TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií

DIPLOMOVÁ PRÁCE

Použití programu GMSH pro vizualizaci výsledků systému Eclipse

Bc. Michal Kolář

Liberec 2011

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: N2612 - Elektrotechnika a informatika

Studijní obor: 1802T007 - Informační technologie

Použití programu GMSH pro vizualizaci výsledků systému Eclipse

Use of the GMSH program for visualization of Eclipse system results

Diplomová práce

Rozsah práce:

Počet stran:	72
Počet obrázků:	33
Počet tabulek:	6
Počet grafů:	2

Autor: Bc. Michal Kolář
Vedoucí práce: doc. Ing. Otto Severýn, Ph.D.

Zadání DP

Jméno a příjmení studenta: **Bc. Michal Kolář**

Vedoucí DP: doc. Ing. Otto Severýn, Ph.D.

Název práce český:

Použití programu GMSH pro vizualizaci výsledků systému Eclipse.

Název práce anglický:

Use of the GMSH program for visualization of Eclipse system results.

Zásady pro vypracování:

1. Seznamte se se systémy Eclipse a GMSH – po uživatelské stránce, především s formáty vstupních a výstupních souborů.
2. Navrhněte a implementujte konvertory mezi výstupy Eclipse a vstupy GMSH (konverze sítí, výsledků výpočtů, příp. materiálových vlastností).
3. Navrhnětě možnosti dalšího propojení mezi programem Eclipse a modelovacími nástroji používanými na NTI (systém Flow123d).
4. Otestujte navržené řešení.

Seznam odborné literatury:

[1] Dokumentace systému Eclipse.

[2] Zdrojové kódy a dokumentace programu GMSH.

Rozsah grafických prací: dle potřeby dokumentace

Rozsah průvodní zprávy: cca. 40 stran

ANOTACE

Tématem této diplomové práce bylo použití programu Gmsh pro vizualizaci výsledků systému Eclipse. Znamená to tedy vytvořit mezikrok v podobě konvertoru, který by dokázal převést výstupní soubory systému Eclipse na vstupní soubory programu Gmsh.

První část objasňuje danou problematiky. První kapitola popisuje simulátor Eclipse, jeho hlavní rysy, využití a formáty některých výstupních souborů, zejména EGRID a PRT. Podobně tak i druhá kapitola popisuje program Gmsh a formát jeho souborů MSH a POS. Následující kapitola obsahuje popis konvertorů egrid2ms, prt100pos a prt300pos, které jsou výsledkem praktické části práce. Čtvrtá kapitola slouží jako uživatelská příručka pro konvertory. Testy konvertorů na různých modelech jsou uvedeny v páté kapitole. Závěrečná kapitola obsahuje shrnutí dosažených výsledků a některé návrhy na možné pokračování práce.

Klíčová slova: Eclipse (ložiskový simulátor), Gmsh, post-processing, konvertor, sítě pro MKP.

ABSTRACT

The goal of this diploma thesis is an application of the program Gmsh as a visualization tool for results of calculations of the Eclipse simulator. For achieving this goal it was necessary to create a convertor transforming output files of the Eclipse to the format of Gmsh's inputs.

The first section brings a light introduction into the field of interest. The first chapter describes the Eclipse simulator, its main features and use and formats of some of the output files, namely EGRID and PRT. In a similar way, the second chapter gives the information about the Gmsh program and format of its files MSH and POS. The following chapter describes the convertors egrid2ms, prt100pos and prt300pos which are the main practical results of the thesis. The fourth chapter is an user guide for the convertors. Tests of the convertors on different models are presented in the fifth chapter. The final chapter contains a summary of the results achieved and some proposals for future development.

Keywords: Eclipse (reservoir simulator), Gmsh, post-processing, convertor, MKP meshes.

PROHLÁŠENÍ

Byl(a) jsem seznámen(a) s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé diplomové práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé diplomové práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Diplomovou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

V Liberci 20.5. 2011

.....

Bc. Michal Kolář

PODĚKOVÁNÍ

Na tomto místě bych rád poděkoval především vedoucímu mé diplomové práce doc. Ing. Ottu Severýnovi, Ph.D. za jeho odbornou pomoc, cenné rady a poskytnuté informace.

Děkuji i své rodině za jejich materiální a duševní podporu a v neposlední řadě přátelům, bez jejichž přispění by práce nevznikla.

Bc. Michal Kolář

Obsah

Seznam obrázků

Seznam tabulek

Seznam grafů

Seznam zkratek

PRESSURE	tlak
SWAT	sycení vody
SGAS	sycení plynu
PERMX	propustnost v ose X
PERMY	propustnost v ose Y
PERMZ	propustnost v ose Z
PORO	porozita
PORV	pórový objem
NTG	net to gross

Úvod

Tématem této diplomové práce je použití programu Gmsh pro vizualizaci výsledků simulací systému Eclipse. Znamená to tedy vytvořit mezikrok v podobě konvertoru, které by dokázal převést výstupní soubory systému Eclipse na vstupní soubory programu Gmsh. Jde tedy o změnu formátu těchto souborů a jejich správnou interpretaci. Důvod vzniku této práce je neexistence rychlého, snadného a nekomerčního software pro vizualizaci výstupů systému Eclipse.

Geofyzikální a geochemické procesy probíhající v přírodě jsou společensky významné (proudění, transport, atd.). Studium těchto procesů je důležité pro možný odhad probíhajících procesů v dané oblasti (PZP, zdroje pitné vody, ukládání radioaktivního odpadu, těžba ropy, atd.). Jedním z konkrétních příkladů je výstavba a provoz podzemních zásobníků plynu situovaných do kompaktních horninových masivů, respektive popis dějů spojených s jejich těžbou.

Jedním z nejdůležitějších fyzikálních dějů v přírodě je tok podzemních vod. S tímto dějem jsou přímo spjaty procesy proudění a transportu, které mohou působit jak současně tak i samostatně bez závislosti jednoho děje na druhém. Dalšími ději jsou sorpční a desorpční procesy, chemické reakce, rozpouštění a srážení minerálních látek, jejichž podstatou jsou vzájemné interakce mezi horninou a proudící tekutinou. Další chemické reakce vznikají přímo v roztoku nosné látky v důsledku jejího míchání a ředění při průchodu prostředím.[3]

Pro simulaci dějů odehrávajících se v horninových masivech je třeba správně pochopit jejich podstatu. Pomocí simulace pak předvídat jejich další vývoj. To ovšem není jednoduchá věc, vezmeme-li v úvahu, že jednotlivé procesy probíhající v podzemí známe, ale jejich popis není jednoduchý. Pro tyto děje většinou na velmi velké oblasti máme pouze bodové informace představované daty získanými z průzkumných vrtů a pomocí geofyziky.

Pro kvantifikaci a pochopení procesů odehrávajících se v podzemí je nezbytným prostředkem matematické modelování. Matematické modely popisují chování podzemní vody, ložisek plynu či ropy, šíření tepla atd. za specifických a předem stanovených podmínek. Na těchto modelech provádíme numerické experimenty. Nadefinujeme vstupní podmínky a zjistíme, jaké odezvy by měl podzemní systém. Nejsou-li některé podmínky známy, nebo neznáme jejich přesnost, lze experimentálně určit míru jejich vlivu na sledované procesy.[3]

K těmto účelům je vhodný systém Eclipse. Jedná se o simulátor vícefázového proudění (např. ropa-voda) používaný pro simulaci úloh těžby ropy a zemního plynu. Možné je však i jeho využití v proudění podzemních vod. Skládá ze simulátoru Eclipse E100, který se specializuje na zjednodušený popis tekutiny. Druhý je Eclipse E300 zaměřený na kompoziční modelování. Pomocí tohoto systému můžeme vytvořit odpovídající model horninového masivu, zadat vstupní a okrajové podmínky a následně pomocí simulace předpovídat chování zkoumaného systému.

Abychom mohli výsledky simulace lépe pochopit, je vhodné je graficky zobrazit. Potřebujeme tedy vhodný software. Gmsh je freewarový program, určený pro grafické zpracování výsledků numerických modelů. Program je díky svým širokým možnostem v nastavení zobrazení naprosto postačující a může konkurovat celé řadě komerčních softwarů. Jeho cílem je poskytnout rychlý, lehký a uživatelsky příjemný nástroj s pokročilými možnosti vizualizace.

Cíle této práce jsou:

- Seznámit se s systémy Eclipse a Gmsh. Jednak po uživatelské stránce, ale především s formáty jejich vstupních a výstupních souborů.
- Navrhnout a implementovat konvertory mezi výstupy Eclipse a vstupy Gmsh (konverze sítí, výsledků výpočtů, případně materiálových vlastností.
- Otestovat navržené řešení.
- Navrhnout možnosti dalšího propojení mezi programem Eclipse a modelovacími nástroji používanými na NTI (systém Flow 123d).

Práce je členěna takto. V první kapitole je popsán systém Eclipse. Najdeme zde také formáty výstupních souborů EGRID a PRT. Druhá kapitola je věnována programu Gmsh, včetně formátu jeho souborů MSH a POS. Třetí kapitola popisuje již zmíněné konvertory egrid→msh a prt→pos. Čtvrtá kapitola obsahuje popis ovládání konvertorů a také potřebného programového vybavení k jejich spuštění. Testy konvertorů na různých modelech včetně jejich popisu nalezneme v kapitole pět. V poslední jsou shrnuty vlastnosti konvertorů, výhody, nevýhody, použití a také případné možnosti rozšíření.

1 ECLIPSE

1.1 Popis Eclipse

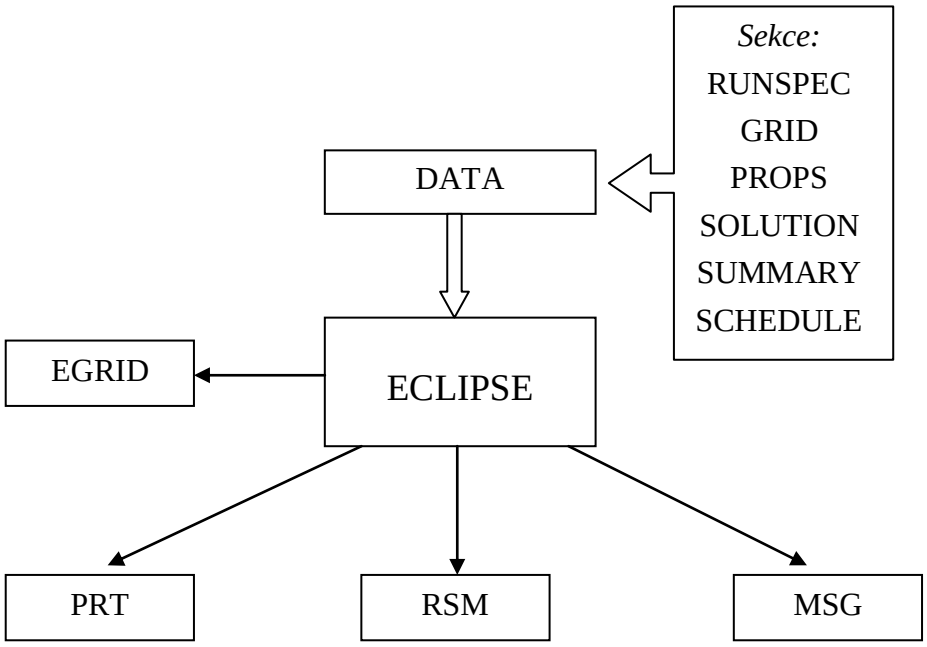
Eclipse je simulátor vícefázového proudění používaný pro simulaci úloh těžby ropy a zemního plynu. Původně vyvinutý firmou ECL (Exploration Consultants Limited) a v současné době vlastněný a na trh uvedený SIS, což je divize společnosti Schlumberger. Jméno Eclipse je zkratka původně používaného názvu ECL Implicit Program for Simulation Engineering.

Jedná se o velmi drahý a složitý komerční software omezený licenční smlouvou. Zejména je určený pro ropné a plynárenské těžební společnosti. Avšak je možné ho využívat i pro modely proudění podzemních vod a další úlohy jako např. geotermika a geomechanika.

Eclipse používá metodu konečných diferencí k řešení parciálních diferenciálních rovnic. Skládá ze dvou samostatných simulátorů. Jedním je Eclipse 100 specializovaný na black oil modelování. Jde o zjednodušený popis tekutiny. Druhý je Eclipse 300 zaměřený na kompoziční modelování, kde tekutina je definována chemickým složením.

Systém Eclipse je napsán v programovacím jazyce Fortran a umožňuje paralelní zpracování složitých simulací. Jelikož používá své vlastní formáty souborů, potřebujeme pro jejich vytvoření, editaci a zobrazení odpovídající software. Přestože je formáty možné editovat v textovém editoru, není tento způsob zrovna ten nejpohodlnější. Dříve bylo možné využít programu Eclipse Office, který umožňoval preprocessing a zobrazení výsledků simulace. Dnes společnost Schlumberger nabízí komplexní řešení v podobě programu *PETREL*.

Petrel je všestranný program, který umí zpracovat danou úlohu od začátku až do konečné fáze (tvorba geologického modelu, vytvoření sítě, spuštění simulace, zobrazení výsledků, …). Nevýhodou je složitost a náročnost programu. Nenajdeme tedy žádný jednoduchý a rychlý software pro zobrazení výsledků simulace.



Obr. 1.1 *Schéma systému Eclipse*

Kde:

DATA

Vstupní soubor systému Eclipse. Obsahuje sekce:

- RUNSPEC – specifikace úlohy, velikosti, gridu
- GRID – definice gridu, propustnosti a porozity
- PROPS – vlastnosti tekutin
- SOLUTION – definice počátečních a okrajových podmínek
- SUMMARY – nastavení výstupů do RSM souboru
- SCHEDULE – scénář průběhu (těžba, vtláčení)

EGRID, PRT, RSM, MSG

Výstupní soubory systému Eclipse. Soubory EGRID a PRT jsou probrány podrobněji v kapitole 1.2 a 1.3.

1.1.1 Výhody a využití systému ECLIPSE

- Nabízí kompletní a robustní sadu numerických řešení pro rychlé a přesné určení dynamického chování kapalin ložisek ropy a zemního plynu.
- Pokrývá celé spektrum simulací.
- Specializuje se na black oil, kompoziční a tepelné simulace ložisek konečné velikosti a také na zjednodušení těchto modelů.
- Výběr je z velké škály možností rozšíření simulátoru pro potřeby uživatele.
- Robustní, rychlý, paralelní a multiplatformní simulační software.

1.2 Výstupní soubor EGRID

Extensible Grid soubor obsahuje definici sítě. Soubory v tomto formátu mají příponu EGRID (binární), nebo FEGRID (textový). Soubor EGRID je generován systémem Eclipse na základě vstupních hodnot v souboru DATA.

Binární soubor EGRID definuje geometrii *sítě* (mřížky, gridu) a pozici všech *buněk* (elementů, prvků) složených z *corner point* („rohových uzlů“). Počet buněk (DIMENS) v každém směru modelu je NX, NY a NZ. Používá se kartézský souřadný systém a počátek je v levém spodním rohu nejvyšší vrstvy. Hloubka v modelu se měří podél osy Z, která je vertikální a směřuje dolů. Větší hodnota značí větší hloubku.

Buňky jsou číslovány nejprve v ose X, poté v ose Y a nakonec v ose Z. Patrné je to z Obr. 1.2. Počet aktivních buněk a jejich posloupnost v tomto souboru se musí shodovat s ostatními přidruženými soubory k dané úloze. Síť může být popsána buď *block centered* nebo *corner point*.

- block centered geometrie

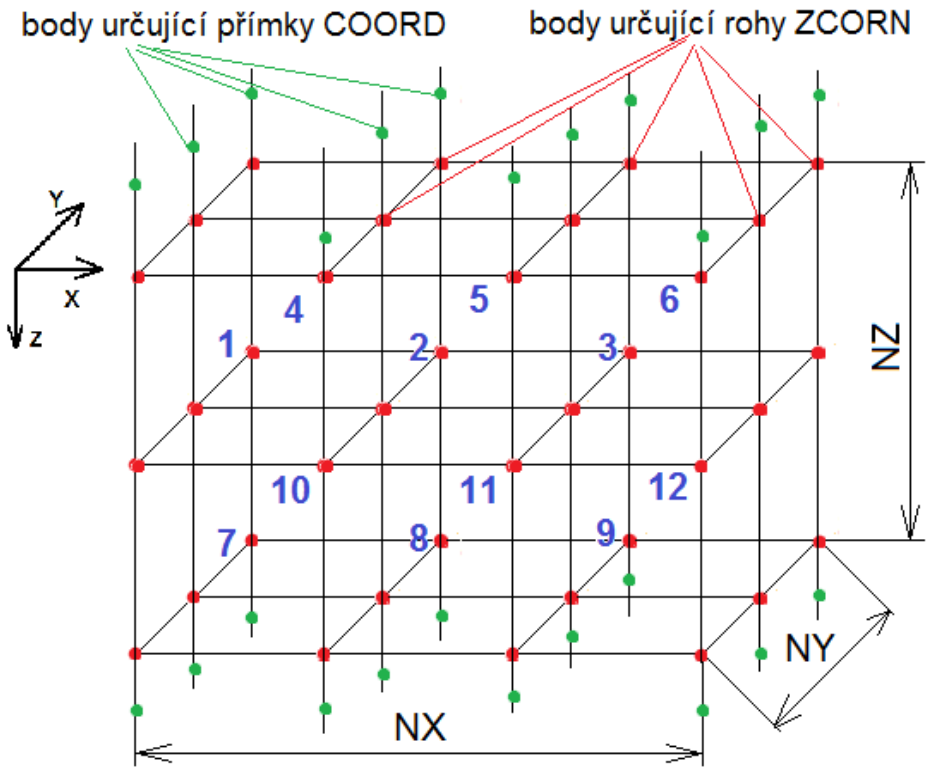
  V případě, že buňky jsou horizontální a všech osm rohů jsou pravé úhly. Každá buňka je definována rozměry tří stran a hloubkou.

1.2.1 Popis sítě typu Corner point geometry

Celá oblast má topologicky podobu kvádru s počtem buněk NXxNYxNZ. Půdorys má NYxNY buněk, tedy (NX+1)x(NY+1) rohů. Každým rohem je vedena přímka, určená dvěma body. Přímky definujeme klíčovým slovem COORD. Uzly rohů buněk leží na příslušných přímkách a jsou určeny Z souřadnicí daného rohu. Z souřadnice definujeme klíčovým slovem ZCORN (Obr. 1.2).

Tato geometrie je definována pomocí šestistěnů, ze kterých je grid složen. Buňky mají různé velikosti a každá je vytvořena pomocí vlastních uzlů. Rohy buňky nemusí svírat pravý úhel. Z toho plyne, že můžeme jednotlivé bloky deformovat, tvořit zlomy a vyklínit vrstvy (Obr. 5.9).

V našem případě budeme uvažovat pouze corner point, protože většina buněk má nepravidelnou strukturu a dokážeme tak vytvořit libovolný grid.

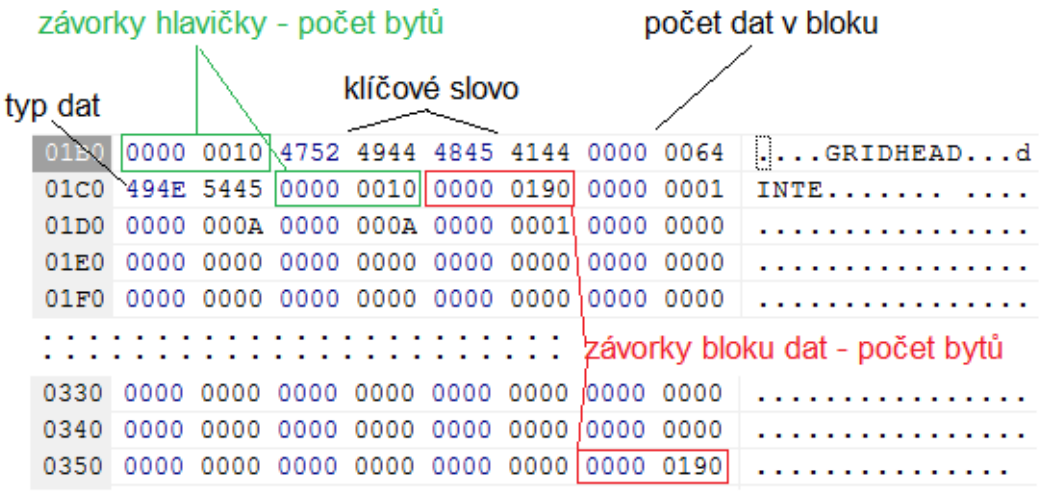


Obr. 1.2 *Ukázka sítě typu corner point*

1.2.2 Formát souboru EGRID

Jedná se o binární soubor, který obsahuje hlavičky. Pomocí hlavičky jsme pak schopni zjistit, jaká data jsou v souboru uložena. Hlavičky uložené v souboru EGRID mají velikost 16B a jsou typu integer. Velikost jednotlivých slov je 32bitů = 4byty.

Každá hlavička popisuje data, která za ní následují. Obsahuje klíčové slovo, počet bytů následujících dat a typ dat. V EGRID se používá uzávorkování bloku dat pomocí hodnoty počtu bytů v bloku. Stejně tak i následující data patřící k této hlavičce jsou uzávorkované pomocí počtu bytů v bloku.



Obr. 1.3 *Ukázka hexa výpisu hlavičky souboru EGRID*

Tabulka 1.1 *Popis FILEHEAD a GRIDHEAD souboru EGRID*

Klíčové slovo	*Počet 4B slov*	*Datový typ*	*Pozice 4B slova*	*Obsah*
FILEHEAD	100	INTE		Hlavička – informace o souboru:
			1.	Číslo verze
			2.	Rok vydání
			3.	Vyhrazeno
			4.	Dřívější kompatibilní verze
			5.	Grid typ:
				0 = corner point
				1 = nestrukturovaný
				2 = hybridní
			6.	Porozita modelu:
				0 = jednoduchá porozita
				1 = dvojitá porozita
				2 = dvojitá propustnost
			7.	Formát původních dat gridu:
				0 = neznámé
				1 = corner point
				2 = block centered
			8.-100.	Nepoužívá se.
GRIDHEAD	100	INTE		Hlavička – informace o gridu:
			1.	Typ gridu:
				0 = kombinovaný
				1 = corner point
				2 = nestrukturovaný
			2.	NX
			3.	NY
			4.	NZ
			5.	Lokální grid:
				0 = globální
				> 0 = lokální
			6.-24.	Nepoužívá se.
			25.	Počet ložisek = 1
			26.	Počet přímek na segment = 1
			27.	Radiální síť:
				0 = neradiální síť
				> 0 = radiální
			28.-100.	Nepotřebné data.

Kde:

FILEHEAD

Obsahuje informace o souboru EGRID. Nejdůležitější data, která musí být v souboru stejně nadefinována, jsou v Tabulka 1.1 podtržena. Především se jedná o typ gridu corner point a jednoduchá porozita.

GRIDHEAD

Obsahuje informace o gridu. Nejdůležitější data, která musí být v souboru stejně nadefinována, jsou v Tabulka 1.1 podtržena. Především se jedná o typ gridu corner point, velikost gridu NX, NY a NZ a také neradiální grid.

Tabulka 1.2 *Popis COORD, ZCORN a ACTNUM souboru EGRID*

Klíčové slovo	*Počet 4B slov*	*Datový typ*	*Obsah*
COORD	6*(NX+1) *(NY+1)	REAL	Hlavička – definice přímek Coordinate lines (přímky) definované dvěma body v prostoru. Jsou zde uloženy body v pořadí X_{1A}, Y_{1A}, Z_{1A}, X_{1B}, Y_{1B}, Z_{1B}, X_{2A}, Y_{2A}, Z_{2A}, X_{2B}, Y_{2B}, Z_{2B}, … Kde: A, B jsou body na přímce 1,2,3,… jsou čísla přímek
ZCORN	8*NX*NY*NZ	REAL	Hlavička – definice Z souřadnic Z souřadnice pro každý uzel v gridu.
ACTNUM	NX*NY*NZ	INTE	Hlavička – definice aktivních buněk Index aktivních buněk: 0 = neaktivní 1 = aktivní

Kde:

COORD

Jedná-li se o geometrii corner point, za klíčovým slovem COORD jsou nadefinovány rohové přímky. Coordinate lines definuje možné pozice bodů gridu, pro každou buňku a pro každé ložisko v modelu. Coordinate lines jsou definovány dvěma trojicemi souřadnic X, Y a Z, které představují dva různé body. Vzhledem k dané hloubce (souřadnice Z) konkrétního bodu buňky a související coordinate lines (přímky) můžeme souřadnice X a Y pro tento bod dopočítat. Jsou-li souřadnice X a Y horního a spodního bodu stejné, potom souřadnice Z zanedbáváme. Souřadnice není nutné přepočítávat a stačí přidat příslušnou Z souřadnici rohu pro konkrétní bod.

ZCORN

Souřadnice Z pro všechny body gridu. Dá se hovořit o hloubce rohů buněk. Každá buňka má 8 rohů (pro každý prvek je definováno 8 vlastních bodů). Tudíž je možné vytvořit buňky různých tvarů. ZCORN obsahuje 8*NX*NY*NZ Z souřadnic pro určení hloubky rohových bodů. Jsou uloženy postupně v ose X vždy pro dva rohy jedné buňky, následuje druhá buňka v ose X atd., poté se přejde v ose Y na další „řádek“ a znovu se čtou souřadnice v ose X. Po načtení vrchních Z souřadnic v jedné vrstvě se přejde na vrstvu spodní a následně na další vrstvy. Číslování buněk je vidět na Obr. 1.2.

ACTNUM

Index aktivních buněk přiřazuje každé buňce v gridu informaci v podobě celého čísla o jeho aktivní činnosti. Hodnota jedna značí, že odpovídající buňka je aktivní, zatímco nula znamená, že je neaktivní. Neaktivní buňka se ve výsledném modelu nezobrazuje a ani při výpočtu není uvažována.

Datový typ INTE

Jde o standardní reprezentaci celých čísel v binární soustavě, které jsou uloženy jako 32bitový single.

Datový tup REAL

Je uložen jako standard IEEE 754 definující binární reprezentaci čísel v pohyblivé řádové čárce v 32bitovém formátu.

1.3 Výstupní soubor PRT

Na rozdíl od EGRID je tento soubor textový. Soubor PRT je standardně hlavní výstup systému Eclipse. Na začátku PRT najdeme zkopírované hodnoty ze vstupního souboru DATA. Každý řádek s takovýmto výpisem začíná číslicí nula a následuje dvojtečka. Následují výpisy hodnot pro celou oblast (PORO, PERMX, PRESSURE, …). Výpis těchto hodnot je řízem pomocí klíčových slov typu RPT v souboru DATA. Tyto výpisy začínají číslicí jedna. V souboru PRT také najdeme různé hlášky programu během simulace.

Ačkoliv je mnoho druhů vypisovaných dat, budeme se zabývat pouze některými. V první řadě se jedná o statická data porozit, propustností a pórového objemu, které jsou nadefinovány v čase nula. Další data se týkají dynamických hodnot tlaku a sycení vodou a plynem, které jsou simulovány a jsou v čase proměnné. Jsou dvě možnosti, jak hodnoty pro jednotlivé buňky následující po těchto klíčových slovech zapsat.

- Konstantní pole

  Je dána pouze jedna hodnota, která je stejná pro všechny aktivní buňky v gridu. Výpis v této podobě je vidět na Obr. 1.4.

```
1                            ********************************************   7
  PORV     AT     0.00   DAYS *         Testovaci uloha Eclipse varianta 1     *
  REPORT   0     1 JAN 2009   *WIN32 RUN                                       *
                              ************************************************

 UNITS    RM3

     CONSTANT ARRAY =   25000.         RM3
```

Obr. 1.4 *Ukázka konstantního pole*

- Trojrozměrná matice

  Hodnoty jsou uloženy do trojrozměrné matice, kde pozice hodnoty v matici odpovídá aktivní buňce v gridu. Sloupce si můžeme představit jako osu X (I), řádky jako osu Y (J). Jelikož není možné vypsat třetí rozměr matice, jsou rozděleny na několik matic, kde každá reprezentuje jednu vrstvu gridu v ose Z (K).

  Je-li v některé hodnotě místo desetinné tečky použita * hvězdička, znamená to, že v daném elementu se nachází sonda.

  Výpis v této podobě vidíme na Obr. 1.5.

```
1                              ******************************************************  13
  PRESSURE AT     1.00   DAYS *         Testovaci uloha Eclipse varianta 1            *
  REPORT   1     2 JAN 2009   *WIN32 RUN                                              *
                              ******************************************************

 UNITS   BARSA

 MINIMUM VALUE =  44.6393  AT (  4,  7,  1)     MAXIMUM VALUE =  55.3640  AT ( 10, 10,  1)

 (I,  J,  K) I=  1       2       3       4       5       6       7       8       9      10

 (*,  1,  1)  50.4290 48.9506 46.7481 45.1276 44.7219 44.6968 44.7105 44.7561 44.8206 44.8552
 (*,  2,  1)  48.5125 48.0049 46.2999 45.0216 44.7059 44.6878 44.7112 44.7770 44.9004 44.9473
 (*,  3,  1)  47.2424 47.0312 45.7937 44.8918 44.6817 44.6659 44.6621 44.8128 45.1172 45.1548
 (*,  4,  1)  46.3596 46.2518 45.3837 44.7838 44.6627 44.6550 44.6769 44.9584 45.4397 45.4898
 (*,  5,  1)  45.7480 45.6851 45.0912 44.7084 44*6547 44.6570 44.7327 45.1946 45.9254 46.0001
 (*,  6,  1)  45.3318 45.2929 44.8947 44.6620 44.6588 44.6725 44.8274 45.5562 46.6446 46.7604
 (*,  7,  1)  45.0549 45.0300 44.7689 44.6393 44.6736 44.7005 44.9707 46.0912 47.6812 47.8760
 (*,  8,  1)  44.8763 44.8585 44.6984 44.6431 44.6946 44.7376 45.1694 46.8399 49.1078 49.4930
 (*,  9,  1)  44.7687 44.7485 44.6983 44.6878 44.7120 44.7749 45.4059 47.7677 50.8971 51.8330
 (*, 10,  1)  44.7213 44.7079 44.6888 44.6901 44.7230 44.8004 45.6004 48.5945 52.6410 55*3640
```

Obr. 1.5 *Ukázka maticového výpisu*

Kde:

PRESSURE

Nasimulovaný tlak ve stanoveném čase pro aktivní buňky modelu. Každé buňce náleží jedno reálné číslo. Jednotky: barsa, psia, atma.

SWAT

Nasimulované sycení vody ve stanoveném čase pro aktivní buňky modelu. Sycení vody můžeme vyjádřit jako poměr objemu vody a objemu pórového prostoru. Platí že SWAT + SGAS = 1. Každé buňce náleží jedno reálné číslo. Jednotky: bezrozměrné.

SGAS

Nasimulované sycení plynu ve stanoveném čase pro aktivní buňky modelu. Sycení plynu můžeme vyjádřit jako poměr objemu plynu a objemu pórového prostoru. Platí že SGAS + SWAT = 1. Každé buňce náleží jedno reálné číslo. Jednotky: bezrozměrné.

PERMX, PERMY, PERMZ

Hodnota propustnosti v ose X, Y a Z pro aktivní buňky modelu. Jedná se o schopnost pórovitého prostředí propouštět kapaliny účinkem hydraulického gradientu nezávisle na jejich druhu a vlastnostech. Každé buňce náleží jedno reálné nezáporné číslo. Nejedná se ovšem o prostupnost mezi buňkami. Jednotky: miliDarcy.

PORO

Hodnota porozity pro aktivní buňky modelu. Porozitu můžeme vyjádřit jako poměr objemu pórů a celkového objemu. Vyjadřuje se v % a běžná hodnota se pohybuje okolo 5 až 25%. Každé buňce náleží jedno reálné nezáporné číslo. Buňky, jejichž objem pórů je nulový, jsou považovány programem jako neaktivní. Jednotky: bezrozměrné.

PORV

Hodnota objemu pórů pro aktivní buňky modelu. PORV většinou nezadáváme, jsme schopni ji dopočítat. Každé buňce náleží jedno reálné nezáporné číslo. Buňky, jejichž objem pórů je nulový, jsou považovány programem jako neaktivní. Klíčové slovo PORV je zcela dobrovolné. Jakékoliv objemy pórů, které se nemění, může program vypočítat z dat GRID. Jednotky: rm^3, rb, rcc.

NTG

Hodnota pro konverzi poměru hustoty z hrubé na čistou pro aktivní buňky modelu. Každé buňce náleží jedno reálné nezáporné číslo. Buňky, jejichž objem pórů je nulový, jsou považovány programem jako neaktivní. Jakékoliv NTG hodnoty, které nejsou uvedeny, jsou nastaveny na výchozí hodnotu 1.0. Jednotky: bezrozměrné.

Kapitola 1 ECLIPSE je upravený a přeformulovaný překlad anglického textu [1].

2 GMSH

2.1 Popis Gmsh

Gmsh je freewarový program, určený pro grafické zpracování výsledků numerických modelů a také pro tvorbu sítí. Program je díky svým širokým možnostem v nastavení zobrazení naprosto postačující a může konkurovat celé řadě komerčních softwarů (např. EnSight a vizualizační prostředky AutoCAD). Byl vyvinutý Christophem Geuzainem a Jeanen-François Remaclem a vydán pod GNU (General Public License).

Jedná se o generátor sítě konečných prvků 2D i 3D s vestavěným CAD engine a post-procesorem. Jeho cílem je poskytnout rychlý, lehký a uživatelsky příjemný nástroj s parametrickým vstupem a pokročilými možnostmi vizualizace. Gmsh je sestaven ze čtyř modulů: *geometry*, *mesh*, *solver* a *post-processing*. Všechny moduly lze ovládat buď interaktivně pomocí grafického uživatelského rozhraní (GUI) nebo pomocí vlastního skriptovacího jazyku. Interaktivní akce generují jednotlivé příkazy v textovém souboru a naopak. To umožňuje automatizovat všechny procedury pomocí smyček, podmínek a volání vnějšího systému.

Vizualizací se rozumí zobrazování skutečnosti, jejichž výsledky jsou graficky znázorněny a vnímány prostřednictvím zrakových receptorů. Vizualizace úzce souvisí s uplatňováním zásady názornosti. Je při tom využíváno metody počítačového modelování.

2.1.1 Geometry

Gmsh reprezentuje spojené plochy pomocí struktury BRep (hraniční reprezentace). Díky tomu můžete spojit všechny druhy speciálních ploch do jediné spojené plochy. Modely jsou vytvořeny metodou shora – dolů postupnou definicí bodů, orientovaných čar (úsečky, kružnice, elipsy, ...), orientovaných ploch (rovinné plochy, přímkové plochy, plochy triangulací, …) a objemů. Složené skupiny z geometrických prvků (tzv. „fyzické skupiny“) můžeme také definovat na základě těchto elementárních geometrických entit. Gmsh je skriptovací jazyk umožňující, aby všechny geometrické prvky byly plně parametrizovány.

2.1.2 Mesh

Mesh (síť), v trojrozměrném prostoru, tvoříme pomocí základních geometrických prvků různých tvarů, které nazýváme elementy. V případě Gmsh se jedná o úsečky, trojúhelníky, čtyřúhelníky, čtyřstěny, hranoly, šestistěny a jehlany. Základní geometrické prvky jsou definovány pouze uspořádaným seznamem svých uzlů. Program Gmsh nám umožňuje vygenerování sítě přímo z geometrie.

Generování sítě se provádí shora-dolů stejně jako u geometrie. Úsečky jsou zpracovány jako první. Z nich je vytvořena síť ploch. Síť ploch se používá pro vytvoření sítě objemů. Dotyk dvou elementů je v jejich vrcholu, na hraně nebo na ploše.

2.1.3 Solver

Externí řešiči mohou být s Gmsh propojeny prostřednictvím Unix nebo TCP/IP připojení. Pomocí řešiče jsme schopni externích výpočtů. Shromažďování a zpracování výsledků je přímo v post-processing modulu. Výchozí řešič propojený s Gmsh je GetDP.

2.1.4 Post-processing

Pojmem post-processing označujeme v kontextu numerického modelování prezentaci výsledků numerických modelů srozumitelnou pro člověka. Soubory post-processingu programu Gmsh mají příponu POS. Jim je přidělena určitá hodnota, která odpovídá velikosti dané fyzikální veličiny, kterou model vypočítal.

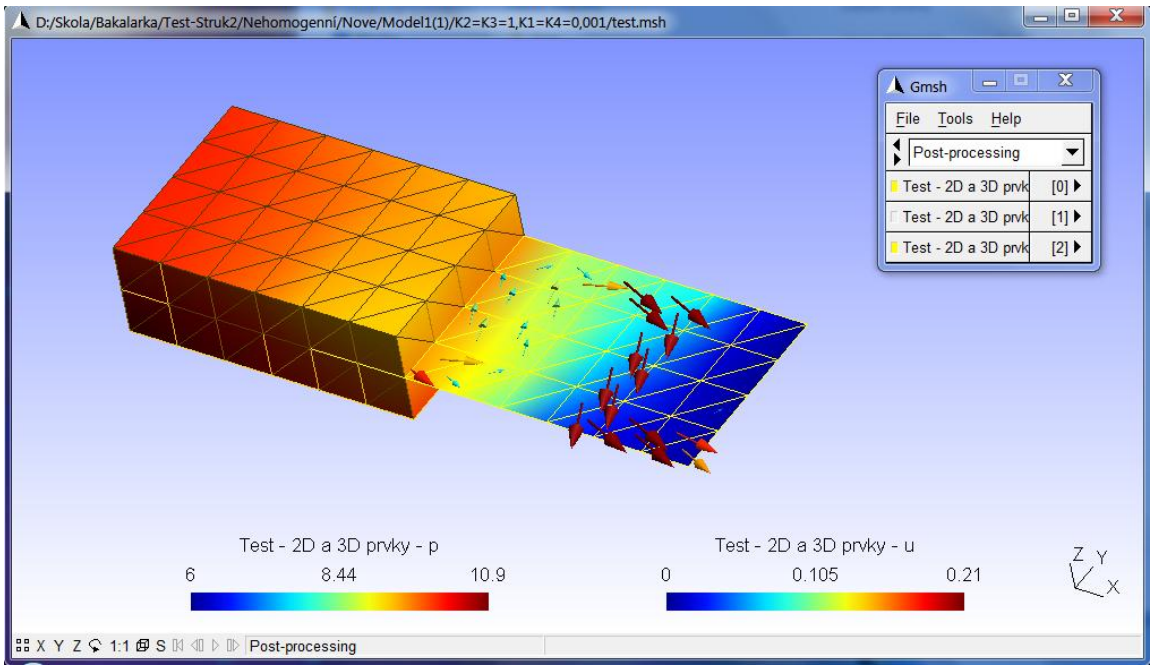
Gmsh může načítat a pracovat se skalárními, vektorovými nebo tenzorovými poli. Skalární hodnoty mohou být vyjádřeny zbarvením modelu. Zatímco vektorová pole jsou zastoupena barevnými trojrozměrnými šipkami. Gmsh nám ale dává více možností zobrazení hodnot výsledků numerických modelů (Tabulka 2.1).

Veškeré nastavení post-processingu je přístupné buď interaktivně, nebo prostřednictvím vstupního souboru v podobě skriptu. Skriptování umožňuje automatizovat všechny post-processingy, jako například vytvoření animace.

2.1.5 Výhody Gmsh

Zde je stručný seznam toho, co Gmsh umí dobře:

- Rychle popsat jednoduché a/nebo „opakované“ geometrie díky uživatelsky definovaným funkcím, smyček, podmínek….
- Vytvářet jednoduché geometrie a sítě pomocí vytažení.
- Komunikovat s externími řešiči, nabízí C, C++, Python… Můžeme přidat další vlastní řešiče a rozhraní.
- Vizualizovat a exportovat výsledky výpočtů na mnoho různých způsobů. Můžeme zobrazovat skalární, vektorové a tenzorové data, provádět různé operace s nimi, exportovat modely do mnoha různých formátů a vytvářet složité animace.
- Běží i na jednoduchých strojích, s i bez grafického rozhraní. Je možné používat Gmsh buď interaktivně, nebo přímo z příkazové řádky.
- Mnoho možností a způsobů konfigurace, které mohou být nastaveny interaktivně pomocí grafického uživatelského rozhraní, příkazové řádky, uvnitř skriptu a pomocí konfiguračního souboru, který si uživatel může nadefinovat podle vlastních potřeb.
- Vše výše uvedené funguje na různých platformách (Windows, Mac a Unix) a zdarma. Pomocí jednoduchého skriptu je to malé, ale výkonné GUI.



Obr. 2.1 *Ukázka programu Gmsh*

2.2 Soubor MSH

Tento soubor slouží k ukládání informací o síti. Jedná se o diskretizaci prostorové oblasti. Skládá se ze dvou částí: v první části jsou uloženy souřadnice uzlů, v druhé pak vlastnosti a posloupnost uzlů definující jednotlivé elementy. MSH formát existuje ve dvou verzích: ASCII a binární. V konvertoru používám ASCII formát. Nevýhodou je velikost souboru. Naopak výhodou je čitelnost souboru pro uživatele, jednoduché úpravy a možnost editace v běžných textových editorech.

MSH v ASCII formátu obsahuje povinný oddíl poskytující informace o souboru ($MeshFormat). Následuje sekce definujících uzly ($Nodes) a elementy ($Elements). Jakýkoliv oddíl s nerozpoznanou hlavičkou, Gmsh ignoruje. Sekce se mohou v jednom souboru opakovat.

2.2.1 Formát souboru MSH

$MeshFormat
version_number file_type
$EndMeshFormat
$Nodes
number_of_nodes
node_number x_coord y_coord z_coord
……
$EndNodes
$Elements
number_of_elements
elm_number elm_type number_of _tags < tag > node_number_list
……
$EndElements

Kde:

version_number

Reálné číslo označující verzi formátu.

file_type

Celé číslo označující typ souboru, v případě ASCII formátu je to nula.

number_of_nodes

Počet všech uzlů v síti definovaných v $Nodes.

node_number

Id číslo n-tého uzlu v síti. Id číslo uzlu musí být kladné (nenulové) celé číslo. Nemusí být nutně řazeny vzestupně nebo na sebe navazovat.

x_coord y_coord z_coord

Tři reálná čísla definující X, Y a Z souřadnice n-tého uzlu v prostoru.

number_of_elements

Počet všech elementů v síti definovaných v $Elements.

elm_number

Id číslo n-tého elementu v síti. Id číslo elementu musí být kladné (nenulové) celé číslo. Nemusí být nutně řazeny vzestupně nebo na sebe navazovat.

elm_type

Celé číslo definující geometrický typ elementu:

1 = úsečka (2 body)

2 = trojúhelník (3 body)

3 = čtyřúhelník (4 body)

4 = čtyřstěn (4 body)

5 = šestistěn (8 bodů)

6 = hranol (6 bodů)

7 = jehlan (5 bodů)

number_of_tags

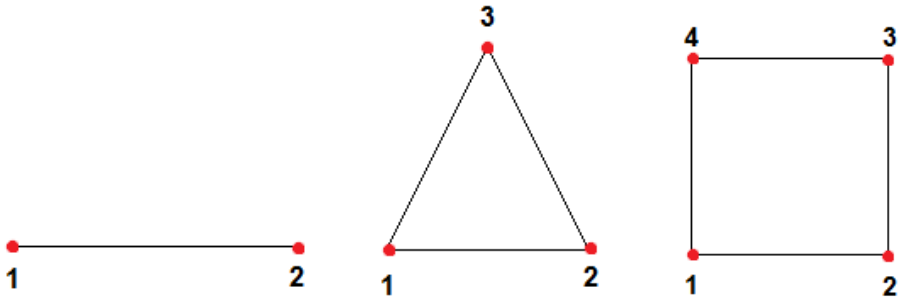
Celé číslo, které udává počet tagů (značek) n-tého elementu následujících hned poté. Význam tagů je na potřebách uživatele. Ve výchozím nastavení je první tag číslo fyzické entity, které elementu patří. Druhý je číslo elementární geometrické entity, které element patří. Třetí je číslo části sítě, které element patří. Značky musí být kladná celá čísla, nebo nula.

node_number_list

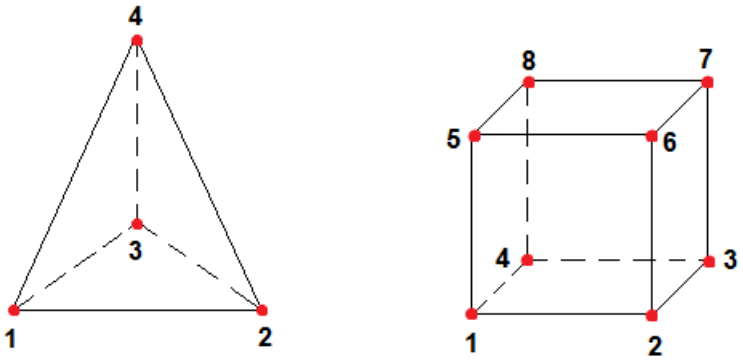
Výčet id čísel uzlů, které tvoří n-tý element. Jejich počet závisí na typu elementu a pořadí je přesně stanoveno. Posloupnost uzlů jednotlivých geometrických typů nalezneme na následujících Obr. 2.2, Obr. 2.3 a Obr. 2.4.

2.2.2 Uspořádání uzlů

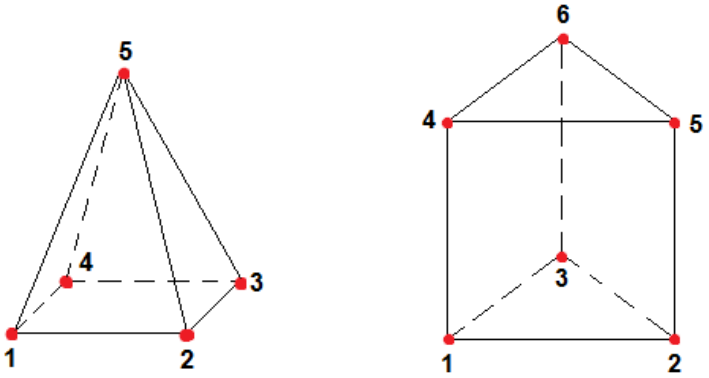
Pro všechny sítě a formáty souborů post-processing jsou referenční elementy definované takto:



Obr. 2.2 *Posloupnost uzlů úsečky, trojúhelníku a čtyřúhelníku*



Obr. 2.3 *Posloupnost uzlů čtyřstěnu a šestistěnu*



Obr. 2.4 *Posloupnost uzlů jehlanu a hranolu*

Krátký příklad:

$MeshFormat	Význam:
2.1 0 8	
$EndMeshFormat	
$Nodes	
6	šest uzlů v síti:
1 0.0 0.0 0.0	uzel #1: souřadnice X, Y a Z [0.0, 0.0, 0.0]
2 1.0 0.0 0.0	uzel #2: souřadnice X, Y a Z [1.0, 0.0, 0.0]
3 1.0 1.0 0.0	atd.
4 0.0 1.0 0.0	
5 2.0 0.0 0.0	
6 2.0 1.0 0.0	
$EndNodes	
$Elements	
2	dva elementy tvořící síť:
1 3 2 99 2 1 2 3 4	element #1: 3 = čtyřúhelník, tag_1 = 99, tag_2 = 2, čísla uzlů 1 2 3 4
2 3 2 99 2 2 5 6 3	element #2: 3 = čtyřúhelník, tag_1 = 99, tag_2 = 2, čísla uzlů 2 5 6 3
$EndElements	

2.3 Soubor POS

Pojmem post-processing označujeme prezentaci výsledků numerických modelů. Post-processing Gmsh je schopen zpracovat více skalárních, vektorových nebo tenzorových hodnot spolu s geometrií sítě. Hodnoty může zadat pomocí několika formátů. Pro naše účely použijeme formát souborů s příponou POS, který je pro člověka snadno srozumitelný. Hodnoty jednotlivých veličiny jsou zde zapsány v číselné podobě.

V terminologii Gmsh každý soubor hodnot nazýváme *View (pole)*. Každé pole má svůj název a můžeme s ním manipulovat buď jednotlivě (každý má svoje vlastní tlačítka v GUI a může na něj být odkazováno podle jeho indexu ve skriptu) nebo globálně. V jednom souboru POS můžeme mít libovolný počet polí.

Ve výchozím nastavení, Gmsh zachází se všemi post-processing poli jako s třírozměrnými grafy, tj. vykresluje skalární, vektorové a tenzorové prvky (body, čáry, trojúhelníky, čtyřúhelníky, atd.) v 3D prostoru. Může také reprezentovat každý post-processing pole obsahující skalární body jako dvourozměrný (X-Y) graf a to buď prostorově, nebo časově orientovaný.

Přestože vizualizace je obvykle interaktivní úloha, Gmsh odhaluje všechny post-processing příkazy a možnosti pro uživatele v jeho skriptovacím jazyce, který umožňuje kompletní automatizaci post-processing procesů.

2.3.1 Formát souboru POS

View "*string*" {
    type (*list_of_coords*) { *list_of_values* };
    …..
};
View "*string2*" {
    type (*list_of_coords*) { *list_of_values* };
    …..
};

Kde:

string

Jméno a identifikátor konkrétního pole. Zadává se do uvozovek.

type

Zkratka typu elementu, který chceme zobrazit a přiřadit mu hodnoty. V Tabulka 2.1 nalezneme všechny zkratky, včetně počtu dalších nutných údajů.

list_of_coords

Obsahuje seznam souřadnic uzlů (X1, Y1, Z1, X2, Y2, Z2, …), které tvoří element daného typu. Počet souřadnic pro konkrétní typ nalezneme v Tabulka 2.1. Uspořádání uzlů tvořících element nalezneme na Obr. 2.2, Obr. 2.3 a Obr. 2.4.

V našem případě jsou souřadnice uzlů elementů v souboru MSH a POS stejné a jedná se i o stejný typ elementů. Tedy tvar, velikost a složení sítě je stejné.

list_of_values

Obsahuje seznam výstupních hodnot pro jednotlivé uzly elementu daného typu. Každý uzel tvořený třemi souřadnicemi má přiřazený počet hodnot podle typu elementu. Pokud bychom zopakovali tento počet hodnot pro jednotlivé uzly, dostali bychom časové kroky výstupních hodnot, které lze spustit jako animaci.

Například, vytvořili bychom element skalární bod, který je tvořen pomocí souřadnic X, Y a Z. Do složených závorek bychom vložili pět výstupních hodnot. Jelikož počet hodnot pro tento element je jedna, dostaneme tak pět časových kroků. Každý krok si můžeme zobrazit postupně, v pořadí v jakém jsou hodnoty uloženy.

Tabulka 2.1 *Značení elementů post-processingu a počet souřadnic*

Element	*Typ*	*Počet souřadnic*	*Počet hodnot*
Bod - skalár	SP	3	1 x počet kroků
Bod - vektor	VP	3	3 x počet kroků
Bod - tenzor	TP	3	9 x počet kroků
Úsečka - skalár	SL	6	2 x počet kroků
Úsečka - vektor	VL	6	6 x počet kroků
Úsečka - tenzor	TL	6	18 x počet kroků
Trojúhelník - skalár	ST	9	3 x počet kroků
Trojúhelník - vektor	VT	9	9 x počet kroků
Trojúhelník - tenzor	TT	9	27 x počet kroků
Čtyřúhelník - skalár	SQ	12	4 x počet kroků
Čtyřúhelník - vektor	VQ	12	12 x počet kroků
Čtyřúhelník - tenzor	TQ	12	36 x počet kroků
Čtyřstěn - skalár	SS	12	4 x počet kroků
Čtyřstěn - vektor	VS	12	12 x počet kroků
Čtyřstěn - tenzor	TS	12	36 x počet kroků
Šestistěn - skalár	SH	24	8 x počet kroků
Šestistěn - vektor	VH	24	24 x počet kroků
Šestistěn - tenzor	TH	24	72 x počet kroků
Hranol - skalár	SI	18	6 x počet kroků
Hranol - vektor	VI	18	18 x počet kroků
Hranol - tenzor	TI	18	54 x počet kroků
Jehlan - skalár	SY	15	5 x počet kroků
Jehlan - vektor	VY	15	15 x počet kroků
Jehlan - tenzor	TY	15	45 x počet kroků
2D text	T2	3	„Text“ x počet kroků
3D text	T3	4	„Text“ x počet kroků

U zobrazení typu 2D text, první dvě čísla v seznamu souřadnic definují souřadnice (X, Y) zobrazeného textu na obrazovce. Počátek je v levém horním rohu okna. Pokud je první (resp. druhý) výraz záporný, pozice se měří od pravého (resp. dolního) okraje okna. Je-li první (resp. druhá) hodnota větší než 99999, textový řetězec je zarovnán horizontálně (resp. vertikálně). Pokud je třetí číslo rovno nule, text je zarovnán vlevo dole a zobrazen pomocí výchozího písma a velikosti. Jinak je třetí hodnota ze seznamu souřadnic převedena na celé číslo, jehož osm nižších bitů znamená velikost písma. Dalších osm bitů definuje font písma (index odpovídá pozici ve fontu v menu GUI) a následujících osm bitů specifikuje zarovnání textu (0 = dole vlevo, 1 = dole uprostřed, 2 = dole vpravo, 3 = nahoře vlevo, 4 = nahoře uprostřed, 5 = nahoře vpravo, 6 = uprostřed vlevo, 7 = střed, 8 = uprostřed vpravo).

3D textový řetězec má první tři čísla v seznamu souřadnic X, Y, Z souřadnici, určující pozici textu v modelu. Čtvrtá hodnota má stejní význam jako třetí číslo u 2D textu.

Pro 2D i 3D textových objekt může seznam výstupních hodnot obsahovat libovolný počet znakových výrazů.

Krátký příklad:

```
View "bod" {
        SP ( 0, 0, 0) { 20 };
};
View "trojuhelnik" {
        ST ( 0,0,0,  5,0,0,  5,5,0 ) { 35, 20, 8 };
};
View "ctyrsten" {
        SS ( 0,0,0,  10,0,0,  5,10,-5,  5,10,5 ) { 30, 20, 25, 10 };
};
```

Kapitola 2 GMSH je upravený a přeformulovaný překlad anglického textu [2].

3 Konvertory výstupů systému ECLIPSE

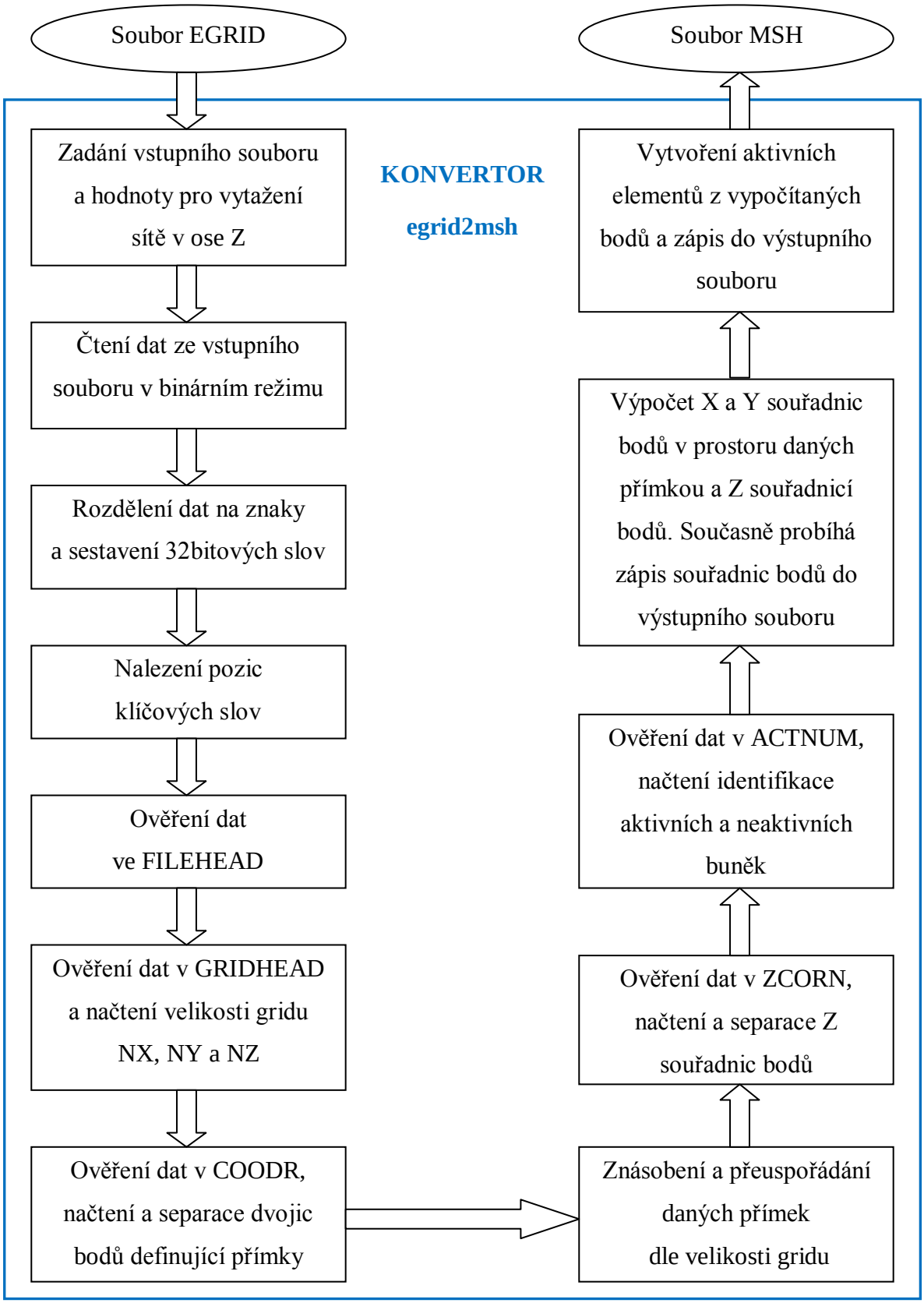
V této kapitole se budu zabývat konvertory výstupu systému Eclipse. Jedná se o transformaci výstupních souborů na soubory kompatibilní s programem Gmsh pro jejich snadné zobrazení. Formáty vstupních a výstupních souborů jsou vysvětleny v kapitolách 1.2 a 2.2. Na přiloženém CD najdeme okomentované zdrojové kódy jednotlivých konvertorů.

Konvertory jsou určeny k převodu výstupního souboru systému Eclipse EGRID na soubor typu MSH, který je možné zobrazit v programu Gmsh jako síť, složenou z jednotlivých elementů. Druhý konvertor převádí výstupní soubor PRT z Eclipse na soubor POS, který lze opět zobrazit v programu Gmsh. Tentokrát dostaneme grafické vyjádření zvolených veličin (PRESSURE, SGAS, SWAT,…) v jednotlivých elementech modelu.

Oba konvertory jsou napsány v programovacím jazyce Perl. Ten byl zvolen s ohledem na zpracovávaná data. U binárního souboru využijeme jednoduché přetypování, naopak u textového souboru rychlé a jednoduché regulární výrazy. Jelikož jejich vstupy jsou pouze jméno vstupního souboru a hodnoty vytažení, spouští se v příkazovém řádku, nebo pomocí vytvořených spustitelných binárních souborů. Grafické rozhraní by zbytečně převod zpomalovalo a nemělo by pro tyto účely žádný efekt. Další nevýhodou grafického rozhraní v jazyce Perl je nutnost mít spolu s *Interpretem* nainstalovány moduly pro GUI. Ty nebývají součástí standardní instalace a musejí se dodatečně nainstalovat. Informace o postupném zpracování souborů se vypisují do okna spuštěného konvertoru, příkazového řádku, konzole atd.

3.1 Konvertor egrid2msh

Nyní bych vysvětlil funkci konvertoru egrid2msh a objasnil převod z EGRID do MSH. Program je členěn na několik samostatných bloků, kde každý provádí určitou operaci s daty a předává je následujícímu bloku. Na Obr. 3.1 je stručně naznačena úprava dat při postupném průchodu jednotlivými bloky konvertoru. Tok dat je ve směru šipek (od zadání vstupního souboru až po výpis do souboru MSH).



Obr. 3.1 *Diagram průchodu dat konvertorem egrid2msh*

Nejdůležitější bloky konvertoru rozeberu podrobněji a pokusím se vysvětlit jejich funkci.

3.1.1 Zadání vstupního souboru a hodnoty pro vytažení sítě v ose Z

Ihned po spuštění programu je nutné zadat jméno vstupního soubor EGRID. Toto jméno souboru se využije v celém konvertoru. Poté uživatel zadá číselnou kladnou hodnotu. Ta slouží k vytažení sítě v ose Z. Tuto funkci využijeme v případě, máme-li příliš plochou síť a potřebujeme zvýraznit členitost modelu (více v kapitole 3.1.10).

Další možnost načtení vstupních hodnot je pomocí spuštění konvertoru s parametry. Pokud se nezadá hodnota vytažení a jméno výstupního souboru, použijí se předdefinované.

3.1.2 Načtení vstupního souboru v binárním režimu, rozdělení na znaky a sestavení 32bitových slov

Pro čtení z binárních souborů není operace *<HANDLE>* vhodná – načítá jeden řádek proměnné délky. Pro čtení zadaného počtu bytů z binárního souboru slouží funkce *read*. Funkce *read* však čte soubory v textovém režimu. Konvertuje znaky konce řádku v souboru ze znaků, které jsou obvyklé v příslušném operačním systému na znak *\n*. Aby čtení probíhalo v binárním režimu, je nutné po otevření souboru provést příkaz *binmode(HANDLE)*.[5]

```
binmode(FILE);
read(FILE, $buffer, $velikost);
```

Nyní načtená data program rozdělí po jednotlivých znacích. Takto rozdělené znaky převádí do číselné podoby v šestnáctkové soustavě. Tím vznikají dvouciferná šestnáctková čísla, která postupně spojuje do čtveřic v pořadí, jak jsou data načítána. Dostáváme tak 32bitová (4B) slova, uložená do pole *@byte*.

```
foreach (split(//, $buffer)) {
    $i += 1;
    $bit = sprintf ("%02x", ord($_));
    $byte[$poz] = "$byte[$poz]" . "$bit";
    if ($i == 5) {
        $poz +=1;
        $i = 1;
        $byte[$poz] = '';
    }
}
```

3.1.3 Nalezení pozic klíčových slov

Jako další krok je nutné najít pozice klíčových slov v jednotlivých hlavičkách souboru EGRID. Konvertor prochází všechna data a podle šestnáctkového vyjádření názvu klíčového slova určí jeho pozici v poli. Klíčová slova jsou důležité pro ověření správného zadání, velikosti, typu a pozice hledaných dat, uložených jako 32bitová slova. Jelikož je vše uloženo v jednom poli *@byte* a známe pozice hlaviček, není problém určit např. pozici souřadnic bodů přímek či velikost gridu.

3.1.4 Ověření dat ve FILEHEAD

Než konvertor začne s jednotlivými výpočty, musí ověřit, zda má model správnou strukturu a může se pokračovat v dalších úpravách dat. Jedná se hlavně o typ gridu *Corner point* a *Single porosity*. Pokud by se jednalo o jiný typ modelu, program se ukončí.

Hlavičky a data ve FILEHEAD jsou uloženy jako integer, proto se použije podprogram *inte* k převodu 32bitových slov do desítkové soustavy a poté se porovná s hodnotami uvedenými v Tabulka 1.1.

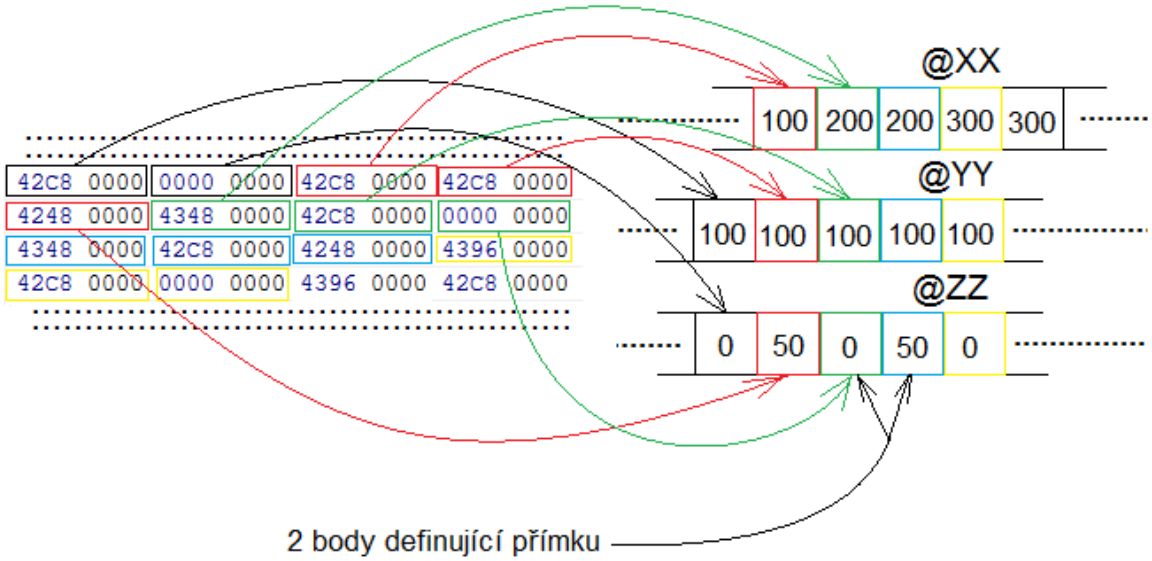
3.1.5 Ověření dat v GRIDHEAD a načtení velikosti gridu NX, NY a NZ

Nejdříve znovu dojde k ověření struktury dat. Opět se musí jednat o grid typu *Corner point* a dále je potřeba určit, zda se nejedná o *Radial* grid. Proběhne-li vše v pořádku, načtou se velikosti gridu NX, NY a NZ. Hodnoty jsou uloženy jako 32bitový integer v šestnáctkové soustavě. K převodu do desítkové soustavy se použije jednoduchý podprogram *inte*, bude probrán v kapitole 3.1.12. Velikost gridu je důležitá pro další výpočty a určení velikosti hledaných dat. Tudíž si uloží do proměnných *$NX, $NY a $NZ.*

3.1.6 Načtení a separace dvojic bodů definující přímky v prostoru

Přímka je definována dvěma body v prostoru. Proto k určení polohy přímky musí program načíst dvojici bodů tj. šest souřadnic. Tyto souřadnice jsou uloženy jako 32bitové realy, zapsané v šestnáctkové soustavě. Formát dat si můžeme prohlédnout v Tabulka 1.2. Pro převod do reálných čísel v desítkové soustavě je napsaný podprogram *real*, který bude probrán v kapitole 3.1.12.

Konvertor opakovaně načítá hodnoty, které rozděluje a ukládá do pole *@XX, @YY a @ZZ*. V poli *@XX* jsou všechny X souřadnice, v *@YY* jsou všechny Y souřadnice a v *@ZZ* jsou všechny Z souřadnice bodů přímek. Vždy dvojice hodnot počítaná od začátku pole, definuje jednu přímku. Při načítání dat musíme dát pozor na hodnoty, které jsou do souřadnic vložené a slouží k rozdělení dat na bloky určité velikosti. Ty však nemají žádnou spojitost se souřadnicemi bodů a musejí být odstraněny.



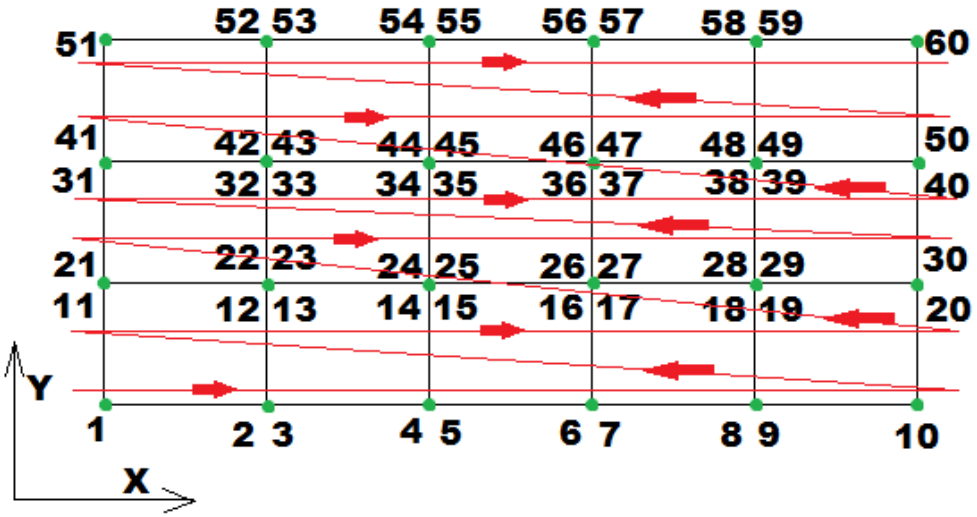
Obr. 3.2 *Postup načítání bodů přímek*

3.1.7 Znásobení a přeuspořádání daných přímek dle velikosti gridu

Pro další výpočty je tento krok velmi výhodný. Jelikož v gridu máme 8xNXxNYxNZ uzlů a přímek je pouze (NX+1)x(NY+1), musíme určité přímky znásobit a vložit na správnou pozici. Algoritmus, který jsem k těmto účelům napsal, prochází souřadnice bodů přímek podle Obr. 3.3.

Corner point definuje pro každý roh elementu vlastní uzly (na Obr. 3.3 tyto uzly vidíme v podobě čísel u rohů elementů). V COORD jsou uloženy rohové přímky, kde však v místě dotyku elementů je definována pouze jedna přímka (zelený bod v rohu elementu). V tomto případě se jedná o 24 přímek, ale 60 rohových uzlů. Proto je lepší kvůli dalším výpočtům mít stejný počet uzlů a přímek.

Jednoduše řečeno, algoritmus zjistí, zda se jedná o přímku v krajních rozích sítě, na stranách sítě, nebo uvnitř sítě. Podle toho ji znásobí tolikrát, kolik je dotyků v příslušném rohu sítě. Dostáváme tedy pro každý uzel sítě vlastní přímku. Průchod sítě je ve směru červených šipek.



Obr. 3.3 *Směr procházení přímek a počet uzlů*

Příklad:

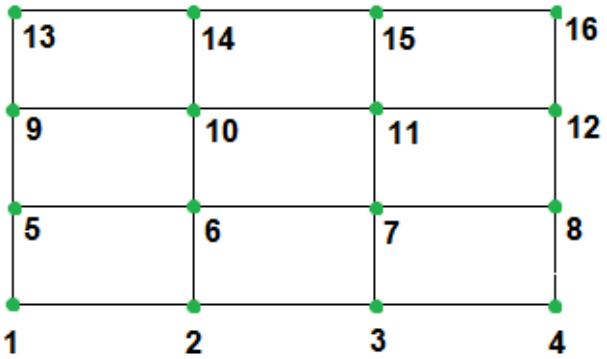
Čísla přímek na Obr. 3.4 jsou:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 (zelený bod)

My však potřebujeme 36 přímek místo 16. Algoritmus znásobí a přeuspořádá potřebné přímky a dostaneme čísla přímek:

1, 2, 2, 3, 3, 4, 5, 6, 6, 7, 7, 8, 5, 6, 6, 7, 7, 8,

9, 10, 10, 11, 11, 12, 9, 10, 10, 11, 11, 12, 13, 14, 14, 15, 15, 16



Obr. 3.4 *Příklad přeuspořádání přímek*

3.1.8 Načtení a separace Z souřadnic bodů

V části ZCORN jsou uloženy Z souřadnice rohů jednotlivých buněk, jejich počet je roven 8xNXxNYxNZ. Každá buňka se skládá z osmi bodů v prostoru. Tyto souřadnice jsou uloženy jako 32bitové realy, zapsané v šestnáctkové soustavě. Pro převod do reálných čísel v desítkové soustavě konvertor použije podprogram *real*, který bude probrán v kapitole 3.1.12.

Program opakovaně načítá hodnoty ze ZCORN a ukládá do pole *@Z*. Při načítání musíme dát opět pozor na hodnoty, které jsou do souřadnic vložené a slouží k rozdělení dat na bloky určité velikosti a musejí být odstraněny.

3.1.9 Načtení identifikace aktivních a neaktivních buněk ACTNUM

ACTNUM nám určuje, zda se jedná o aktivní, nebo neaktivní buňku. Pokud je buňka aktivní, zahrnuje se do výpočtu a ve výsledku se i zobrazuje. V opačném případě se s danou buňkou nepočítá a ani by se neměla objevit ve výsledném gridu. Hodnoty jsou uloženy jako integer a nabývají hodnot 0 při neaktivní buňce a 1 při aktivní buňce. Jejich počet je NXxNYxNZ.

Tyto informace konvertor načítá do pole *@act* a uloží počet aktivních buněk. Při načítání musí znovu odstranit hodnoty, které jsou do dat vložené a slouží k rozdělení dat na bloky určité velikosti.

3.1.10 Výpočet X a Y souřadnic bodů v prostoru daných přímkou a Z souřadnicí

Body, které definují jednotlivé buňky, jsou zadány přímkou a Z souřadnicí. Ty jsou již načtené, rozdělené a uložené ve čtyřech polích *@XX, @YY, @ZZ* a *@Z*. Teď jen zbývá podle jednoduchého vzorce dopočítat hodnoty X a Y souřadnic bodů, známe-li Z souřadnice ze ZCORN. K výpočtu se použije vzorec (1). Vypočítané hodnoty jsou poté zaokrouhleny.

$$\frac{X-X_1}{X_2-X_1}=\frac{Y-Y_1}{Y_2-Y_1}=\frac{Z-Z_1}{Z_2-Z_1} \qquad (1)$$

kde:

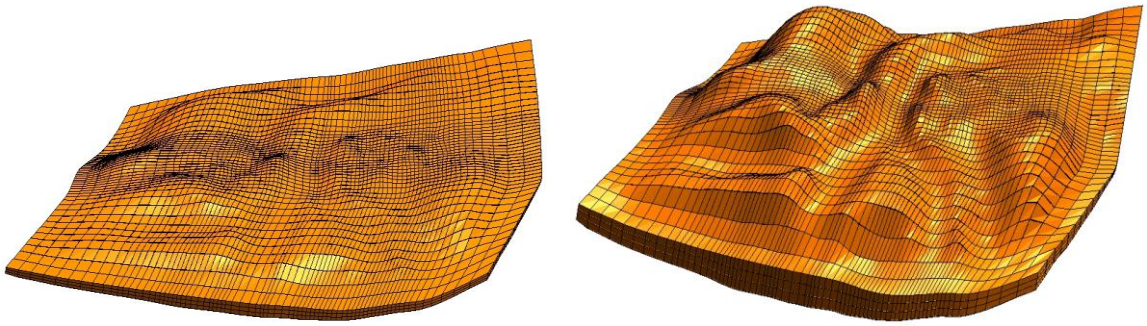
X_1, Y_1, Z_1…… souřadnice prvního bodu přímky

X_2, Y_2, Z_1…… souřadnice druhého bodu přímky

X, Y, Z …… souřadnice libovolného bodu na přímce

Nyní využijeme vytažení v ose Z, které jsme zadávali při spuštění programu. V podstatě se jedná o vynásobení Z souřadnice námi zadanou hodnotou. Pokud bychom zadali hodnotu vytažení = 1, nedojde k ovlivnění původního tvaru modelu. Hodnota menší než 1 model sníží, naopak hodnota větší než 1 model zvýrazní. Příklad je na Obr. 3.5. Jelikož osa Z směřuje ve vertikálním směru dolu, nikoli nahoru, musíme ještě hodnoty Z souřadnic vynásobit hodnotou -1. Tím dosáhneme změny směru osy Z.

Vypočítané souřadnice rovnou zapisuji do výstupního souboru MSH ve formátu uvedeném v kapitole 2.2.

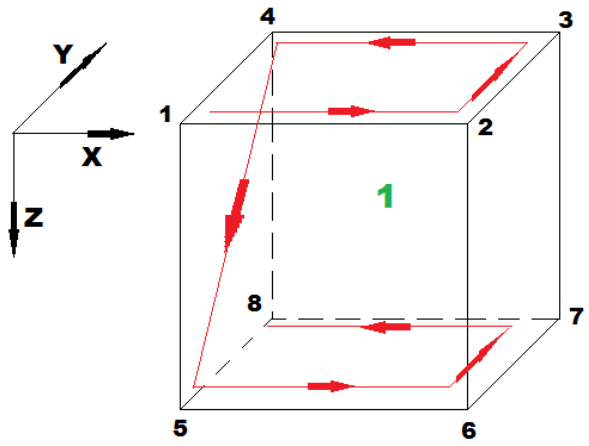


Obr. 3.5 *Původní síť (vlevo) a vytažená síť (vpravo)*

3.1.11 Vytvoření elementů z vypočítaných bodů a zápis do výstupního souboru

V posledním bloku program složí z vypočítaných bodů šestistěny. Každý element se skládá z osmi bodů v prostoru, které jsou pro něj jedinečné. Musí tedy procházet všechny body a určit, ke kterému elementu patří a na jaké pozici se bude nacházet. Vše je znovu závislé na velikosti modelu. Příklad složení elementů je na Obr. 3.7.

V případě, že se jedná o aktivní buňku, což zjistí z ACTNUM, zapíší se tyto body v daném pořadí do výstupního souboru MSH pod souřadnice bodů, ze kterých jsou elementy složeny. Na Obr. 3.6 je vidět posloupnost jednotlivých bodů, jež jsou nutné ke složení jednoho elementu.



Obr. 3.6 *Pořadí uzlů v šestistěnu*

Příklad:

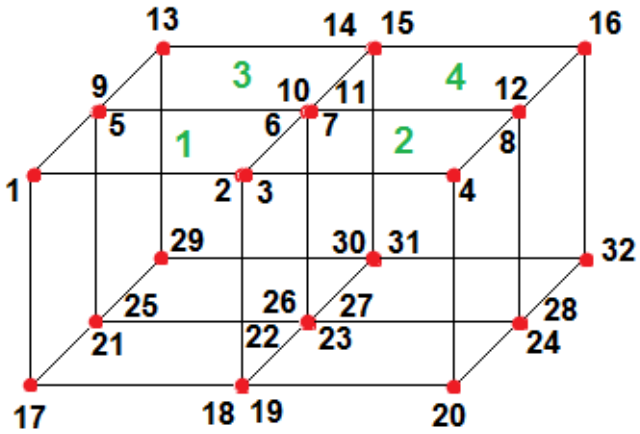
Na Obr. 3.7 je síť složená ze 4 elementů, tedy 32 rohových uzlů.

Element 1 se skládá z uzlů: 1, 2, 6, 5, 17, 18, 22, 21

Element 2 se skládá z uzlů: 3, 4, 8, 7, 19, 20, 24, 23

Element 3 se skládá z uzlů: 9, 10, 14, 13, 25, 26, 30, 29

Element 4 se skládá z uzlů: 11, 12, 16, 15, 27, 28, 32, 31



Obr. 3.7 *Příklad složení sítě*

3.1.12 Převod 32bitového čísla v šestnáctkové soustavě typu real a integer

Konvertor osahuje i dva podprogramy. Jeden slouží k převodu proměnné typu integer v šestnáctkové soustavě, druhý pak převádí proměnné typu real do desítkové soustavy. K tomuto účelu je zde použito procedur *PACK* a *UNPACK*. Ty jsou vhodné při převodu mezi textovou a binární reprezentací dat.

Podprogram se volá klíčovými slovy *inte* a *real.* V závorce je hodnota, kterou chceme převést. V tomto případě se jedná o osmimístné šestnáctkové číslo.

První podprogram převede hexa řetězec na ASCII a následně na celé číslo v desítkové soustavě. Druhý převede hexa řetězec na převrácený ASCII a následně tuto hodnotu převede na desítkové číslo s desetinou čárkou.

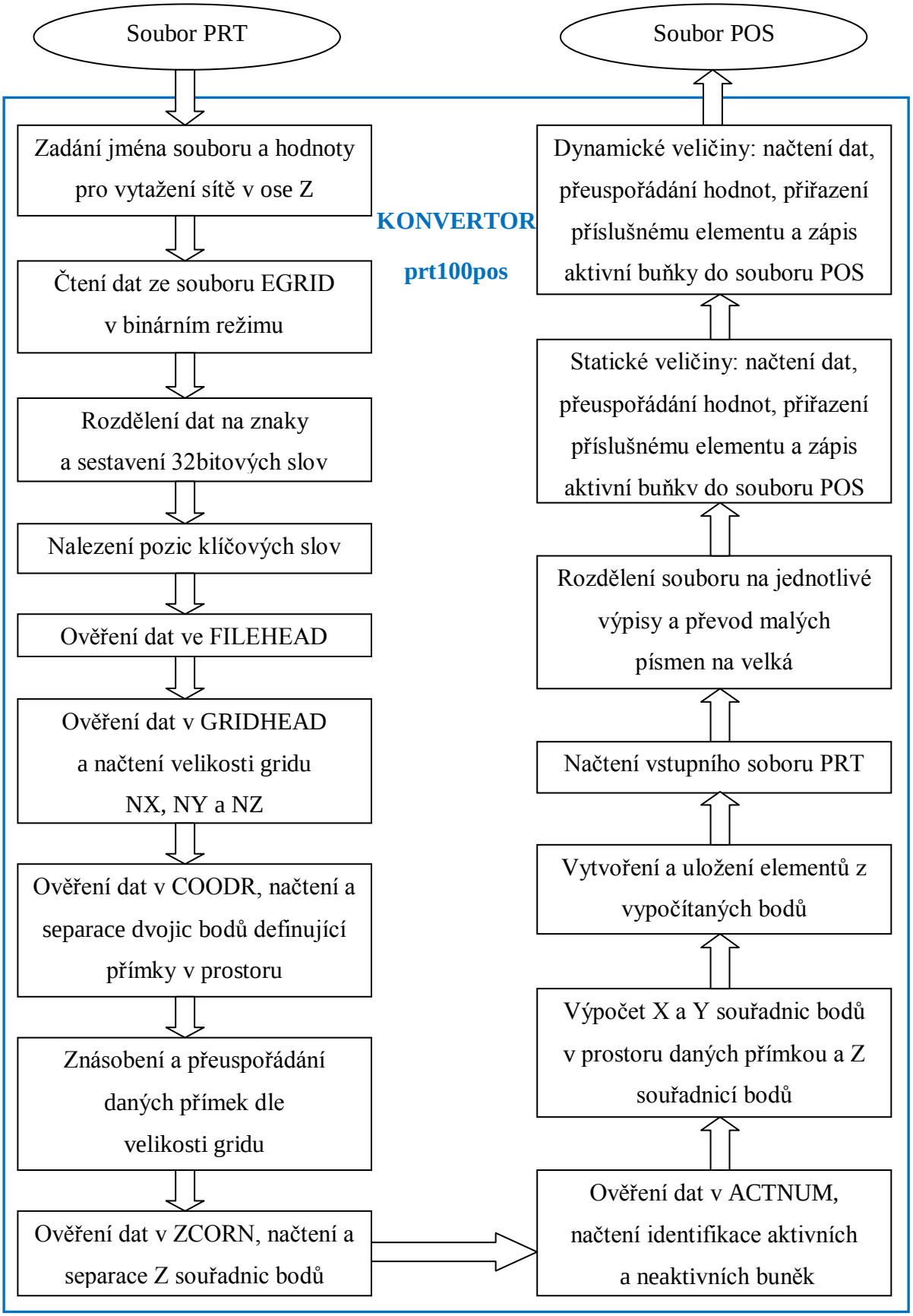
```
sub inte{  my($hodnota) = @_;
            return unpack ("N", pack "H8", $hodnota); }
sub real{  my($hodnota) = @_;
           return unpack ("f", reverse pack "H8", $hodnota); }
```

3.2 Konvertor prt100pos

Nyní popíšeme konvertor výstupního souboru PRT. Ten obsahuje data uložená jako text. Jde o převod ze souboru PRT na soubor POS. Program je znovu členěn na několik samostatných bloků, kde každý provádí určitou operaci s daty a předává je následujícímu bloku. Číslo 100 v názvu konvertoru znamená, že se jedná o převod z Eclipse E100.

Abychom mohli graficky zobrazit výstupní hodnoty (PRESSURE, SWAT, SGAS,…) do zadané oblasti, konvertor musí nejdříve zpracovat soubor EGRID. Tím dostává souřadnice bodů v prostoru, ze kterých jsou jednotlivé elementy složeny. Ze souboru PRT získá hodnoty výstupních veličin, které přiřadí odpovídajícím elementům. Výsledkem je grafické znázornění výstupních veličin v celé síti.

V následujících bodech je naznačena úprava dat při postupném průchodu jednotlivými bloky konvertoru. Na Obr. 3.8 můžeme vidět jednotlivé bloky konvertoru. Tok dat je ve směru šipek (od zadání vstupního souboru až po výpis do souboru POS).



Obr. 3.8 *Diagram průchodu dat konvertorem prt100pos*

Nejdůležitější bloky konvertoru rozeberu podrobněji, vytvoření sítě nalezneme v kapitole 3.1.

3.2.1 Načtení souboru EGRID a vytvoření modelu sestaveného ze šestistěnů

Bloky konvertoru, které zpracovávají vstupní soubor EGRID jsou stejné, jako u konvertoru egrid2msh v kapitole 3.1. Jedná se o načtení vstupního souboru, uložení jako 32bitová slova, separace bodů přímek a jejich přeuspořádání, separace Z souřadnic bodů a výpočet souřadnic bodů daných přímkou a Z souřadnicí.

Malá změna je pouze v uložení souřadnic bodů. Vše plyne z formátu souboru POS. Ukládají se souřadnice bodů do pole v přesně určené posloupnosti, ve které definují jednotlivé šestistěny. Neukládají se tudíž souřadnice bodů jak jdou za sebou. Jim je pak přiřazena hodnota dané veličiny.

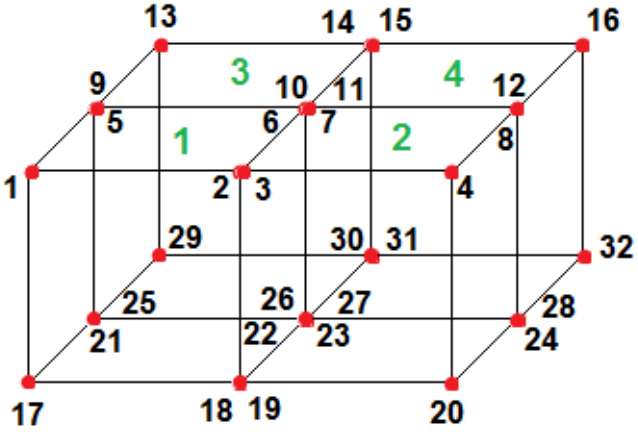
Příklad:

Na Obr. 3.9 je síť složená ze 4 elementů, tedy 32 rohových uzlů.

@Element1 se skládá ze souřadnic:

$X_1, Y_1, Z_1, X_2, Y_2, Z_2, X_6, Y_6, Z_6, X_5, Y_5, Z_5, X_{17}, Y_{17}, Z_{17}, X_{18}, Y_{18}, Z_{18}, X_{22}, Y_{22}, Z_{22}, X_{21}, Y_{21}, Z_{21}$

@Element2 se skládá ze souřadnic: $X_3, Y_3, Z_3, X_4, Y_4, Z_4, \ldots\ldots\ldots\ldots\ldots\ldots$



Obr. 3.9 *Příklad složení sítě*

3.2.2 Načtení vstupního souboru PRT

Jakmile jsou vytvořeny jednotlivé elementy, načte se soubor PRT. Ten je na rozdíl od souboru EGRID textový a nemá úplně definovanou strukturu. Celý text konvertor uloží do jednoho pole *@text*, kde každý prvek pole obsahuje právě jeden řádek vstupního souboru.

3.2.3 Rozdělení souboru na jednotlivé výpisy a převod malých písmen na velká

Přestože není formát dat přesně definovaný, jsou zde použita klíčová slova, která se v určité struktuře opakují. Podle nich je program schopen text rozdělit na jednotlivé kroky simulace. Kvůli vyhledávání klíčových slov se převedou všechna písmena na velká a vyhneme se tak případným nejasnostem v textu.

Dílčí výpis začíná vždy číslem 1 hned na začátku řádku a končí číslem výpisu. Této vlastnosti jsem využil při dělení textu. Data zapsaná mezi dvěma jedničkami jsou uložena jako jeden prvek v poli *@bloky*. Na obrázku (Obr. 3.10) je vidět jeden výpis ze souboru PRT, což je jeden prvek pole.

```
1                            ***************************************************** 13
  PRESSURE AT     1.00   DAYS *         Testovaci uloha Eclipse varianta 1          *
  REPORT   1     2 JAN 2009   *WIN32 RUN                                           *
                              *****************************************************

 UNITS   BARSA

 MINIMUM VALUE =  44.6393  AT (  4,  7,  1)     MAXIMUM VALUE =  55.3640  AT ( 10, 10,  1)

 (I,  J,  K) I=  1       2       3       4       5       6       7       8       9      10

 (*,  1,  1)  50.4290 48.9506 46.7481 45.1276 44.7219 44.6968 44.7105 44.7561 44.8206 44.8552
 (*,  2,  1)  48.5125 48.0049 46.2999 45.0216 44.7059 44.6878 44.7112 44.7770 44.9004 44.9473
 (*,  3,  1)  47.2424 47.0312 45.7937 44.8918 44.6817 44.6659 44.6621 44.8128 45.1172 45.1548
 (*,  4,  1)  46.3596 46.2518 45.3837 44.7838 44.6627 44.6550 44.6769 44.9584 45.4397 45.4898
 (*,  5,  1)  45.7480 45.6851 45.0912 44.7084 44.6547 44.6570 44.7327 45.1946 45.9254 46.0001
 (*,  6,  1)  45.3318 45.2929 44.8947 44.6620 44.6588 44.6725 44.8274 45.5562 46.6446 46.7604
 (*,  7,  1)  45.0549 45.0300 44.7689 44.6393 44.6736 44.7005 44.9707 46.0912 47.6812 47.8760
 (*,  8,  1)  44.8763 44.8585 44.6984 44.6431 44.6946 44.7376 45.1694 46.8399 49.1078 49.4930
 (*,  9,  1)  44.7687 44.7485 44.6983 44.6878 44.7120 44.7749 45.4059 47.7677 50.8971 51.8330
 (*, 10,  1)  44.7213 44.7079 44.6888 44.6901 44.7230 44.8004 45.6004 48.5945 52.6410 55.3640
```

Obr. 3.10 *Ukázka jednoho prvku pole @bloky*

3.2.4 Statické vlastnosti modelu – PERMX, PERMY, PERMZ, PORO, PORV, NTG

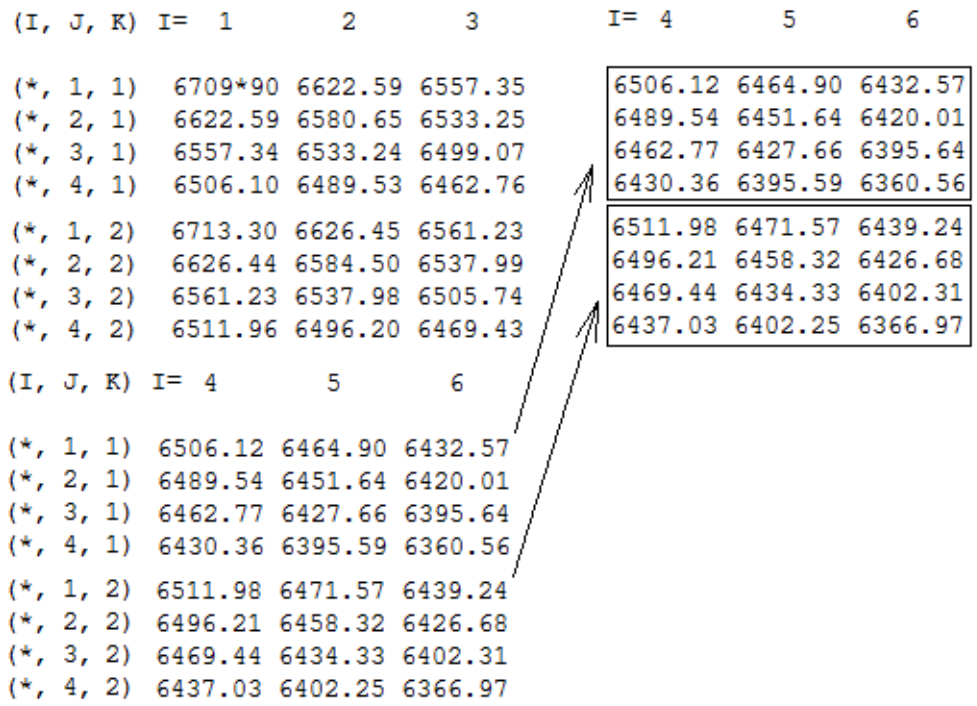
Obsahují-li některé z výpisů uložených v poli *@bloky* údaje o statických vlastnostech modelu, provede se jejich načtení a uložení do souboru POS. Nejprve je potřeba zjistit, jak jsou tyto hodnoty zapsány. Jedna z možností je konstantní pole (Obr. 1.4) a druhá maticový zápis (Obr. 1.5).

Pokud jsou hodnoty uloženy jako konstantní pole, znamená to, že všechny elementy v síti mají stejnou hodnotu výstupní veličiny. Stačí tuto hodnotu uložit do výstupního pole tolikrát, kolik je elementů v síti.

V případě maticového zápisu program najde začátky řádků matice a načítá prvky do jednoho pole po řádcích. Jelikož se jedná o 3D model, musela by mít matice tři rozměry. Další problém je ve velikosti matice. Do jednoho řádku je možné zapsat pouze 15 hodnot. Kvůli zápisu je rozdělena na několik matic, kde každá reprezentuje jednu vrstvu modelu v ose Z a počet prvků v řádku je vždy maximálně 15. Nastane-li situace, kdy je počet hodnot v jednom řádku větší než 15, musí se jednotlivé matice přeuspořádat a poskládat „za sebe“. Zjednodušeně řečeno, proházet všechny řádky matice uložené po jednotlivých prvcích v poli a vložit na správnou pozici (Obr. 3.11).

Příklad:

Jedná se o zjednodušený příklad, kde uvažuji pouze řádky délky 3 prvky. Vlevo vidíme matici uloženou v PRT, vpravo pak přesun částí matice na správnou pozici.



Obr. 3.11 *Ukázka přeuspořádání matice*

Jsou-li data uložená, stačí je vypsat spolu se souřadnicemi odpovídajících elementů do souboru POS ve formátu uvedeném v kapitole 2.3. Zápis jednotlivých elementů do souboru provádíme pouze tehdy, jedná-li se o aktivní buňku. Také konvertor přidá údaj o jednotce zobrazené veličiny. Každá tato statická vlastnost je uložena jako samostatné pole (*View)* v jednom souboru.

3.2.5 Tlak v modelu – PRESSURE, saturace vody – SWAT, saturace plynu – SGAS

Postup při načítání hodnot PRESSURE, SWAT a SGAS je téměř shodný jako u statických vlastností modelu. Nejprve se ověří pomocí klíčových slov, zda v datech jsou tyto hodnoty vůbec zaznamenány a v případě jejich existence je potřeba zjistit, jak jsou zapsány. Jedna z možností je konstantní pole (Obr. 1.4) a druhá maticový zápis (Obr. 1.5).

Hodnoty uložené jako konstantní pole, nebo v podobě vícerozměrné matice mají stejnou strukturu jako u statických vlastností. Podrobnosti najdeme v kapitole 3.2.4.

Soubor PRT zpravidla obsahuje více než jeden výpis simulace obsahující tyto dynamické data. Proto tento postup získávání hodnot opakuji, dokud neprojdu všechny výpisy. Takto uložené hodnoty vkládám za sebe do příslušného prvku pole.

Jsou-li data uložená, stačí je vypsat spolu se souřadnicemi odpovídajících elementů do souboru POS ve formátu uvedeném v kapitole 2.3. Zápis jednotlivých elementů do souboru se provádí pouze tehdy, jedná-li se o aktivní buňku. Dále se přidá údaj o jednotce zobrazené veličiny a vloží se i údaje o čase, ve kterém jsou hodnoty nasimulovány. Pro každou veličinu PRESSURE, SWAT a SGAS je vytvořen samostatný soubor POS, ve kterém je jedno pole (*View*) obsahující několik hodnot ke každému elementu. Výsledkem je animace zobrazující jednotlivé kroky simulace v čase.

3.3 Konvertor prt300pos

V této práci jsem se měl zabývat konvertory systému E100, ale menší úpravou jsem vytvořil i zjednodušené konvertory pro systém E300. Znovu se jedná o transformaci vstupních a výstupních souborů na soubory kompatibilní s programem Gmsh pro jejich snadné zobrazení. Konvertor je napsán v programovacím jazyce PERL. Z důvodů uvedených v kapitole 3 je opět bez přítomnosti GUI.

Jde o převod výstupního souboru systému E300 EGRID na soubor typu MSH, který je možné zobrazit v programu Gmsh jako síť, složenou z jednotlivých elementů. Tento převod probíhá úplně stejně jako u E100, proto je konvertor egrid2msh systému Eclipse E100 a E300 stejný. Více informací nalezneme v kapitole 3.1.

Druhý konvertor převádí výstupní soubor PRT z E300 na soubor POS, který lze opět zobrazit v programu Gmsh. Tentokrát dostaneme grafické vyjádření zvolených veličin (PRESSURE, SGAS, SWAT,…) v jednotlivých elementech modelu.

3.3.1 Rozdíl mezi prt100pos a prt300pos

Konvertor výstupu systému Eclipse E300 je téměř shodný se systémem E100. Ten je popsaný a vysvětlený v kapitole 3.2. Odlišnost je ve způsobu rozdělení dat v podobě textu na jednotlivé výpisy. Také nalezení konkrétních výpisů obsahujících hledané statické a časově závislé hodnoty se provádí na jiném principu.

Každý výpis je oddělen řádkem obsahující pomlčky a končící číslem výpisu na rozdíl od konvertoru E100, kde řádek začíná číslem 1, následují * a končí číslem výpisu. Dále nemůžeme použít klíčová slova, jelikož se v souboru PRT vyskytují i mimo hlavičku. Konvertor hledá dané výpisy pomocí celého názvu veličiny. Největší odlišnost je tedy v hlavičce jednotlivých výpisů. Hodnoty jsou zde uloženy stejně jako u E100 v podobě konstantního pole a vícerozměrné matice. Načítání veličin probíhá tudíž stejně (kapitola 3.2.4).

```
1                              ********************************************************  13
  PRESSURE AT     1.00   DAYS *         Testovaci uloha Eclipse varianta 1             *
  REPORT   1     2 JAN 2009   *WIN32 RUN                                               *
                               ************************************************************

 UNITS   BARSA
```

```
------------------------------------------------------------------------------------------------------------94
 PRESSURE Barsa         Oil phase pressures                      4.01000   Days  report step 5, 22 Feb 2011
 Run on  5/03/2011 at 09:30:59 version 2009.2   cpu 4.47 elapsed 5.00 memory      1 Mb
 KARTEZSKY model v E300 geomechanika blackoil
------------------------------------------------------------------------------------------------------------
```

Obr. 3.12 *Porovnání hlaviček E100 (horní) a E300 (dolní)*

4 Ovládání konvertorů

Konvertory jsou napsány v jazyce Perl. Jedná se o interpretovaný programovací jazyk, který je multiplatformní a spustíme ho prakticky ve všech operačních systémech. Aby bylo možné konvertory spustit, musí být v počítači nainstalován interpret. Ověřit je to možné pomocí příkazu *perl –v.* Případně si ho můžeme stáhnout např. z internetu (*www.cpan.org*) a doinstalovat.[5]

Další možností jak spustit konvertory, je použití vytvořených binárních souborů, přeložených pomocí programu *Perl2Exe* ze skriptů konvertorů. Překladač může nahradit interpret, neboť ho ke své činnosti nepotřebuje. Výsledkem je tedy binární soubor, který obsahuje všechny potřebné knihovny a může být šířen na jakýkoliv systém.

Výhody jsou následující:

- zdrojový kód zůstává skrytý
- program funguje nezávisle na přítomnosti interpretu v systému
- snadné spuštění programu v libovolném systému
- binární konvertory jsou rychlejší než při použití interpretu

Nevýhody jsou pak:

- soubor v binární podobě je větší oproti skriptu, který má pouze pár kB
- program není možné doladit nebo provést úpravy zdrojového kódu

Co se týče programu Gmsh, je zdarma dostupný na internetu na stránce *geuz.org/gmsh/*. Najdeme zde různé verze programu v podobě binárních spustitelných souborů a pro různé platformy. Také máme možnost stažení zdrojových kódů, referenčního manuálu a jiné dokumentace, ukázkových skriptů a také si prohlédnout screenshoty.

Gmsh se skládá z jednoho spustitelného souboru. Není potřeba žádné instalace. Pouze pro lepší spouštění skriptů je dobré mít přidružené typy souborů MSH a POS k programu.

Všechny konvertory a program Gmsh nalezneme na přiloženém CD.

4.1 Spuštění konvertorů

Ovládání konvertorů je velmi jednoduché. Máme několik možností, jak programy spustit. První možností je spuštění konkrétního skriptu (*egrid2msh.pl, prt100pos.pl* a *prt300pos.pl*) přímo v operačním systému, jako spustitelný soubor. Musíme mít však nainstalovaný interpret a přidružené soubory *.pl k programu Perl. Další možností je použití příkazového řádku nebo konzole. V tomto případě spustíme skript příkazem *perl jméno_konvertoru.pl*. Opět musí být v systému nainstalovaný interpret. Třetí variantou jak spustit konvertor je použití přeložených binárních souborů *egrid2msh.exe, prt100pos.exe* a *prt300pos.exe* . Tato možnost je ze všech nejlepší. Jednak není potřeba interpret, program funguje v libovolných systémech a zpracování souborů je rychlejší.

Konvertory můžeme spustit také s parametry. První parametr je jméno, případně celá cesta k vstupnímu souboru. Jméno zadáváme bez přípony souboru. Druhý parametr je hodnota vytažení. Jde o nepovinný údaj. Nezadáme-li žádnou hodnotu, nastaví se vytažení na jedna. Chceme-li použít i třetí parametr, musíme tuto hodnotu zadat. Poslední parametr je jméno výstupního souboru, případně celé cesty a jména výstupu. Opět se jméno zadává bez přípony, ta je doplněna automaticky. Tento parametr je také nepovinný. Pokud bychom nic nezadali, výstup se nastaví do adresáře se vstupním souborem a jméno bude stejné, jako je jméno vstupního souboru . Zadáme-li pouze jméno výstupního souboru (bez cesty), uloží se do pracovního adresáře s konvertorem.

Parametry se zadávají:

perl jméno_konvertoru.pl vstupní_soubor hodnota_vytažení výstupní_soubor

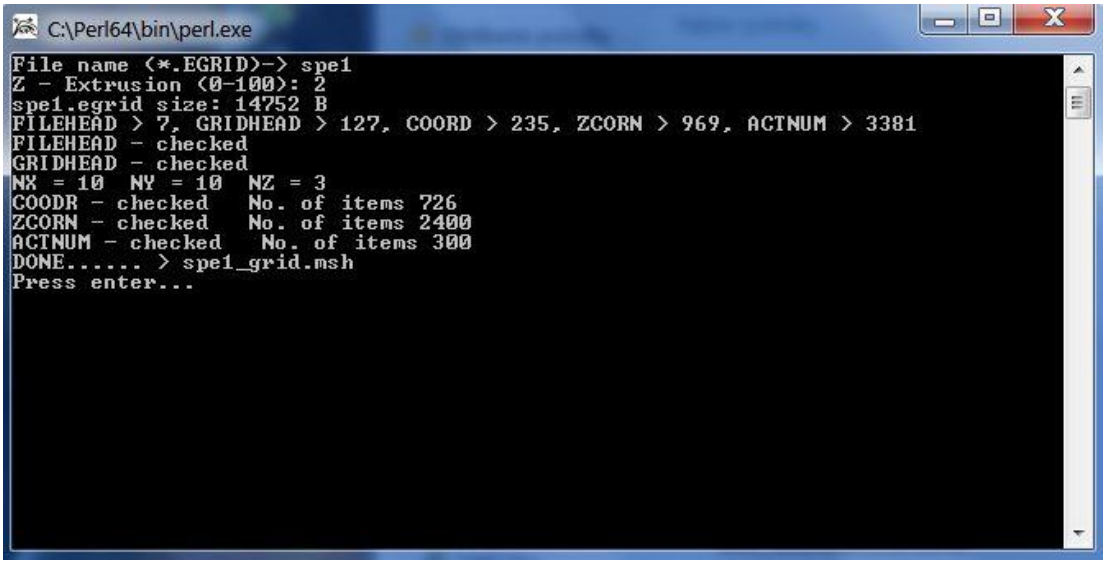
jméno_konvertoru.exe vstupní_soubor hodnota_vytažení výstupní_soubor

4.2 egrid2msh

Po spuštění konvertoru, ať už v podobě perlovského skriptu, nebo binárního spustitelného souboru, jsme vyzváni k zadání vstupního souboru EGRID. Výhodné je vložit konvertor přímo do adresáře s tímto souborem. Stačí zadat pouze jméno souboru bez přípony, ta je doplněna automaticky. Pokud je konvertor v jiném adresáři, zadáme cestu k vstupnímu souboru a jeho jméno bez přípony. Volbu potvrdíme klávesou Enter. To však neplatí, pokud je konvertor spuštěn s parametrem. Zadané parametry se načtou jako vstupní hodnoty a konvertor spustí převod.

Nyní jsme tázáni na hodnotu vytažení v ose Z. Vstup je omezen na kladné reálné číslo. Zadáme-li hodnotu 1, dostaneme původní model. Hodnotou v rozmezí 0 až 1 model snížíme, naopak číslem větším než 1 zvýšíme. Tato funkce je vhodná pro zvýraznění příliš „plochých“ modelů. Vloženou hodnotu potvrdíme stiskem klávesy Enter.

Zadáme-li jméno souboru a hodnoty vytažení, konvertor začne zpracovávat vstupní soubor EGRID. Průběh zpracování se vypisuje do spuštěného okna. Jako první se zobrazí velikost vstupního souboru. Následují pozice klíčových slov v binárním souboru EGRID. Pokud úspěšně proběhne ověření FILEHEAD a GRIDHEAD, zobrazí se hláška o kontrole a velikost gridu NX, NY a NZ. Dále se zobrazí hlášení o ověření COORD, ZCORN a ACTNUM včetně počtu prvků jim příslušejících. V případě chybného zadání v hlavičkách se konvertor ukončí a vypíše danou nesrovnalost. Proběhne-li celý převod v pořádku, vypíše se na obrazovku jméno souboru MSH, který se nachází ve stejném adresáři jako vstupní soubor EGRID. Nyní stačí ukončit konvertor stiskem Enter a otevřít soubor MSH v programu Gmsh. Průběh zpracování můžeme vidět na obrázku (Obr. 4.1).



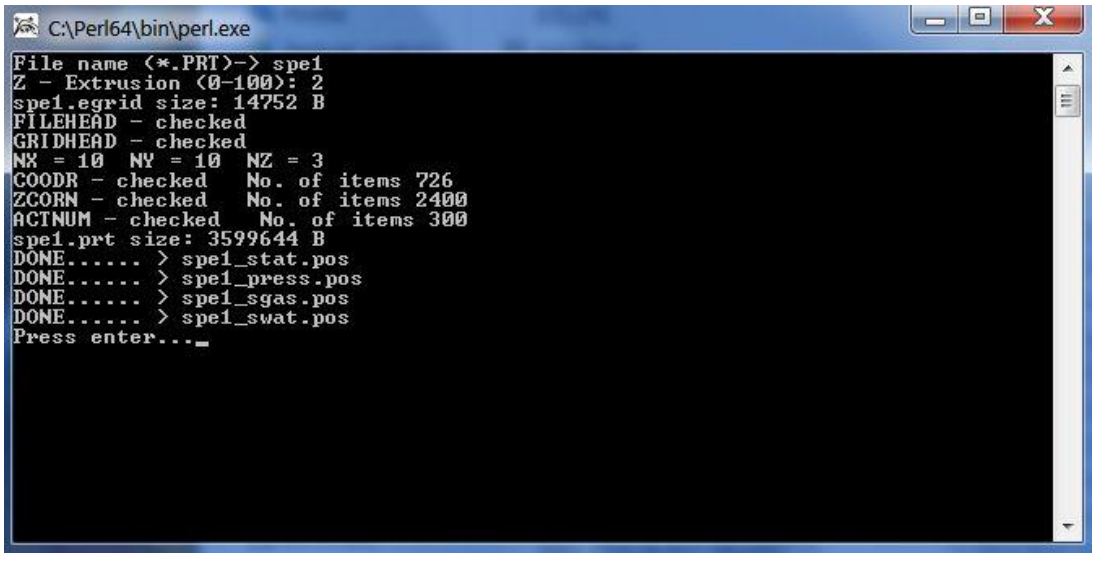
Obr. 4.1 *Náhled konvertoru egrid2msh*

4.3 prt100pos a prt300pos

Konvertory na převod vstupního souboru EGRID a PRT jsou po uživatelské stránce téměř shodné. Po spuštění konvertoru jsme vyzváni k zadání jména souboru, tentokrát však výstupního souboru systému Eclipse PRT. Převádíme-li soubor PRT, musíme mít ve stejném adresáři a se stejným jménem i soubor EGRID. Z něho se generuje síť modelu. Jméno souboru se opět zadává bez přípony. Volbu potvrdíme klávesou Enter. To však neplatí, pokud je konvertor spuštěn s parametrem. Zadané parametry se načtou jako vstupní hodnoty a konvertor spustí převod.

Jako další vložíme hodnotu vytažení v ose Z, stejně jako tomu je v kapitole 4.2. Zadáme-li jméno souboru a hodnoty vytažení, konvertor začne zpracovávat vstupní soubor EGRID. Průběh zpracování se vypisuje do spuštěného okna. Jako první se zobrazí velikost vstupního souboru. Pokud úspěšně proběhne ověření FILEHEAD a GRIDHEAD, zobrazí se hláška o kontrole a velikost gridu NX, NY a NZ. Dále se zobrazí hlášení o ověření COORD, ZCORN a ACTNUM včetně počtu prvků jim příslušejících. V případě chybného zadání v hlavičkách se konvertor ukončí a vypíše danou nesrovnalost.

Po načtení dat ze souboru EGRID konvertor přistoupí ke čtení dat ze vstupního souboru PRT. Nejdříve se zobrazí velikost právě načítaného souboru. Existují-li výpisy obsahující hodnoty o statických datech, PRESSURE, SWAT a SGAS, je o tom uživatel informován výpisem na obrazovku. Také je informován o názvu souborů a jejich umístění (standardně adresář se zpracovávanými soubory). Nyní stačí ukončit konvertor stiskem Enter a otevřít soubory POS v programu Gmsh. Průběh zpracování je na Obr. 4.2.



Obr. 4.2 *Náhled konvertoru prt100pos*

5 Testování navržených konvertorů

V této kapitole si představíme několik modelů, které byly pomocí konvertoru převedeny a zobrazeny v programu Gmsh. Jedná se o tři skupiny testů. První jsou testovací úlohy. Druhá a třetí skupina jsou výsledkem měření reálné oblasti podzemních zásobníků plynu. Všechny modely jsou k dispozici na přiloženém CD.

Při vytváření konvertoru jsem pro první testy použil nejmenší síť *Test.* Je vhodná pro odladění konvertoru, protože má malý počet uzlů, elementů a pravidelnou strukturu. Dá se tedy jednoduše překontrolovat soubor MSH a POS. Jako další model jsem použil *Lobodice* a nakonec nejsložitější model *Tvrdonice.*

Testovací model: TEST

Problémy: Načtení binárního souboru v šestnáctkové soustavě a převod do desítkové.

Sestavení rohových uzlů (jedna přímka definuje více rohových uzlů).

Řešení: Nalezení potřebných informací na www.cpan.org a sestavení podprogramů.

Vytvoření algoritmu pro znásobení a přeuspořádání přímek pro každý rohový bod.

Testovací model: LOBODICE

Problémy: Nejen hlavičky a data jsou uzávorkovány pomocí velikosti bytů v bloku, ale i samotná data tyto hodnoty obsahují (v případě většího množství dat).

Řešení: Odstranění nežádoucích hodnot z dat.

Testovací model: TVRDONICE

Problém: Zkreslení a deformace sítě, vycházely nesmyslné modely.

Řešení: Násobení Z souřadnic rohových bodů hodnotou vytažení, až po výpočtu souřadnic bodů X a Y.

U funkčních konvertorů jsem provedl ještě pár malých úprav. Výstupní soubory konvertoru MSH a POS jsou textové, tudíž každý nadbytečný znak znamená zvětšování souboru. Proto jsem vymazal všechny nepotřebné mezery a zaokrouhlil souřadnice bodů na tři desetinná místa. Jelikož je v některých sítích i 500tis. uzlů neboli 1,5mil. souřadnic, dosáhl jsem tak značného zmenšení velikosti výstupních souborů. Porovnání vidíme v Tabulka 5.2 a Tabulka 5.3.

5.1 Testovací modely

Tato skupina testů byla vytvořena za účelem otestovat systém Eclipse. Byly vytvořeny jednoduché čtvercové gridy s různým počtem elementů. Na tyto jednoduché modely jsem konvertory začal programovat. Jejich geometrie a velikost byla vhodná pro počáteční odladění konvertorů. Jsou to modely s názvem GASWATER (Obr. 5.1), SPE1(Obr. 5.2) a TEST (Obr. 5.3).

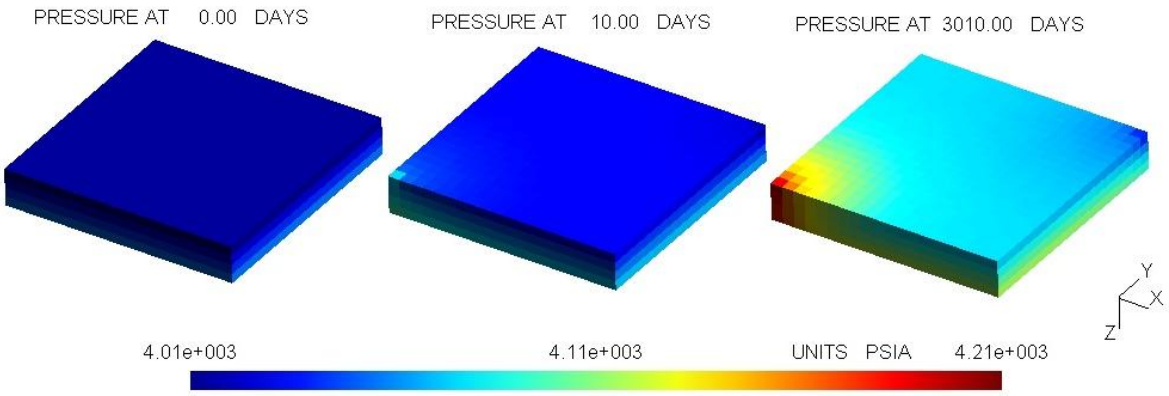
Na obrázcích můžeme vidět průběh simulace tlaku a sycení vody v daném čase. Ve skutečnosti se jedná o animaci v programu Gmsh, zobrazující změnu těchto veličin v čase. Jelikož má animace více jak 50 hodnot simulace, ukážu pouze pár mezikroků.

- GASWATER

  Velikost sítě: 20 x 20 x 4

  Počet uzlů: 12 800

  Počet elementů celkem / aktivních: 1 600 / 1 600



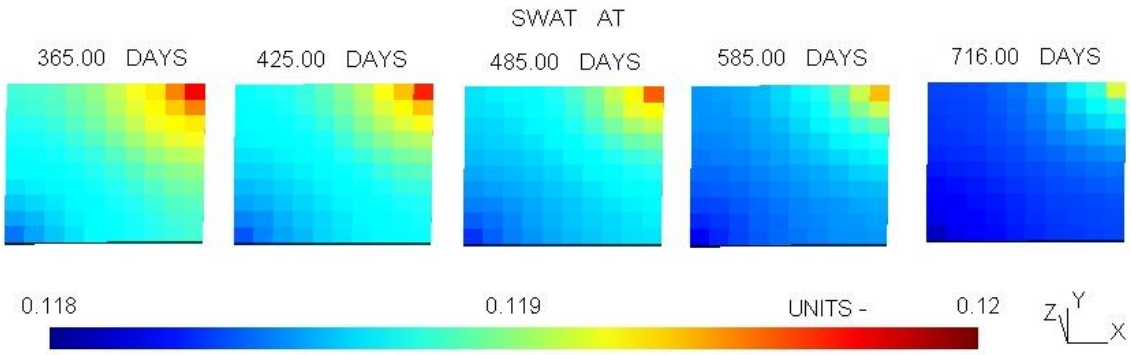
Obr. 5.1 *Gaswater - průběh tlaku v síti*

- SPE1

  Velikost sítě: 10 x 10 x 3

  Počet uzlů: 2 400

  Počet elementů celkem / aktivních: 300 / 300



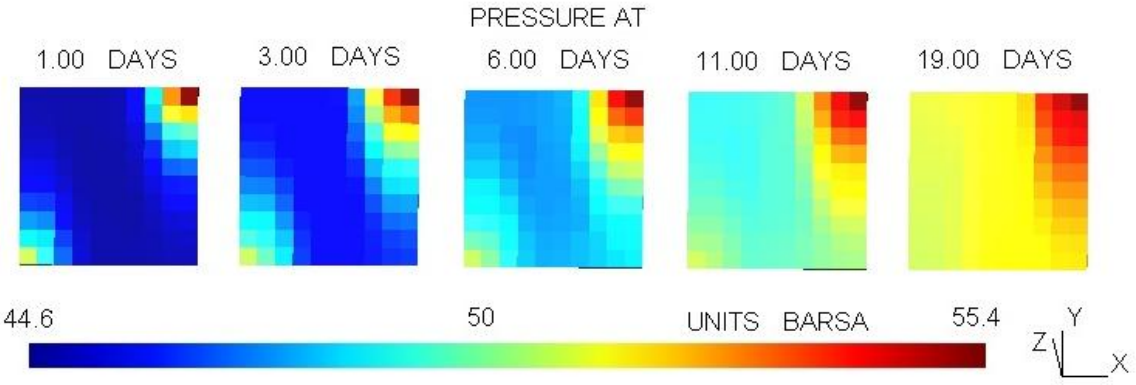
Obr. 5.2 *Spe1 - průběh saturace vody v síti*

- TEST

  Velikost sítě: 10 x 10 x 1

  Počet uzlů: 800

  Počet elementů celkem / aktivních: 100 /100



Obr. 5.3 *Test - průběh tlaku v síti*

5.2 Ploché modely

Jak už název napovídá, jedná se o spojité modely s nevýraznou strukturou povrchu. Ačkoliv na obrázcích je vidět nerovný povrch modelu, ve skutečnosti není až tak členitý. Pro lepší zobrazení a představu o tvaru modelu jsem použil vytažení v ose Z, které je v konvertoru implementováno. Modely v této části jsou výsledkem měření reálné oblasti. Jedná se o tři podzemní zásobníky plynu – Dolní Dunajovice (Obr. 5.4), Lobodice (Obr. 5.6) a Štramberk (Obr. 5.5).

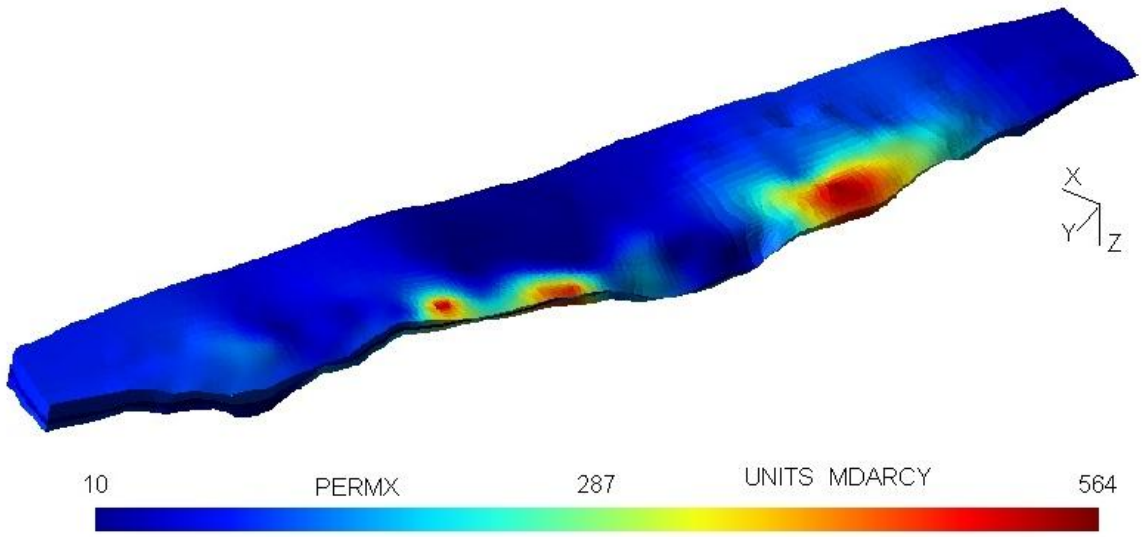
Na rozdíl od předchozích modelů, kde můžeme vidět dynamické veličiny, následující tři obsahují informace o statických veličinách. Ty jsou měřeny v čase nula na začátku simulace.

- DUNAJOVICE – dd

  Velikost sítě: 24 x 158 x 3

  Počet uzlů: 107 008

  Počet elementů celkem / aktivních: 13 376 / 13 376



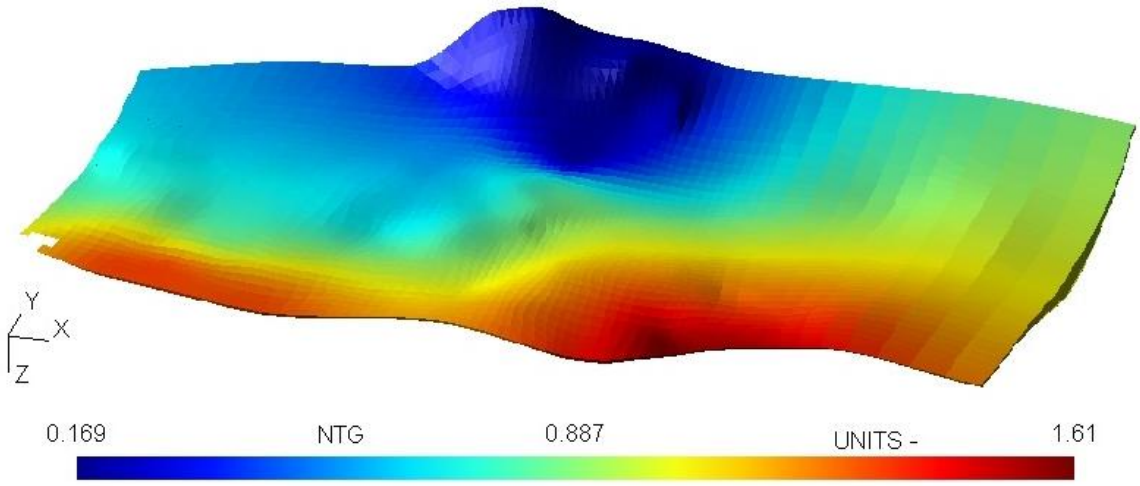
Obr. 5.4 *Dunajovice - propustnost sítě*

- ŠTRAMBERK – st

  Velikost sítě: 73 x 49 x 6

  Počet uzlů: 171 696

  Počet elementů celkem / aktivních: 21 462 / 15 974



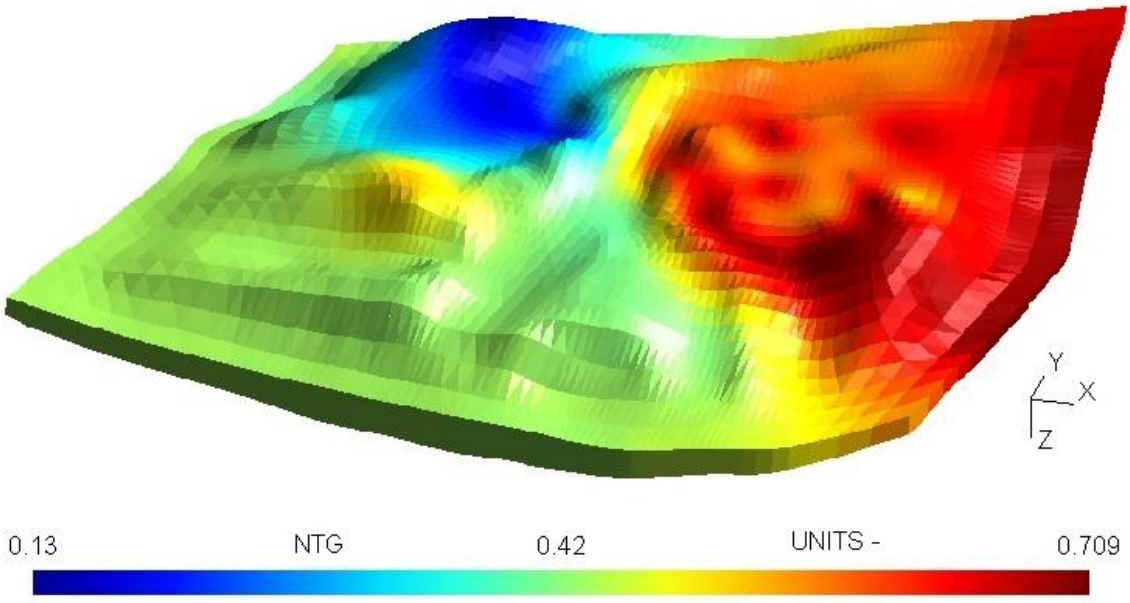
Obr. 5.5 *Štramberk - NTG sítě*

- LOBODICE – lb

  Velikost sítě: 76 x 44 x 4

  Počet uzlů: 107 008

  Počet elementů celkem / aktivních: 13 376 / 13 376



Obr. 5.6 *Lobodice - NTG sítě*

5.3 Nespojité modely

Nyní se podíváme na nejsložitější gridy ze všech testovaných. Je to model podzemního zásobníku plynu Tvrdonice. Skládá se ze tří samostatných částí: 8. sarmatský obzor, 12.-14. sarmatský obzor a 9. bádenský obzor. Celý model složený z jednotlivých částí vidíme na Obr. 5.9. Jednotlivě je pak 8. sarmatský a 9. bádenský obzor zobrazen na Obr. 5.7 a Obr. 5.8.

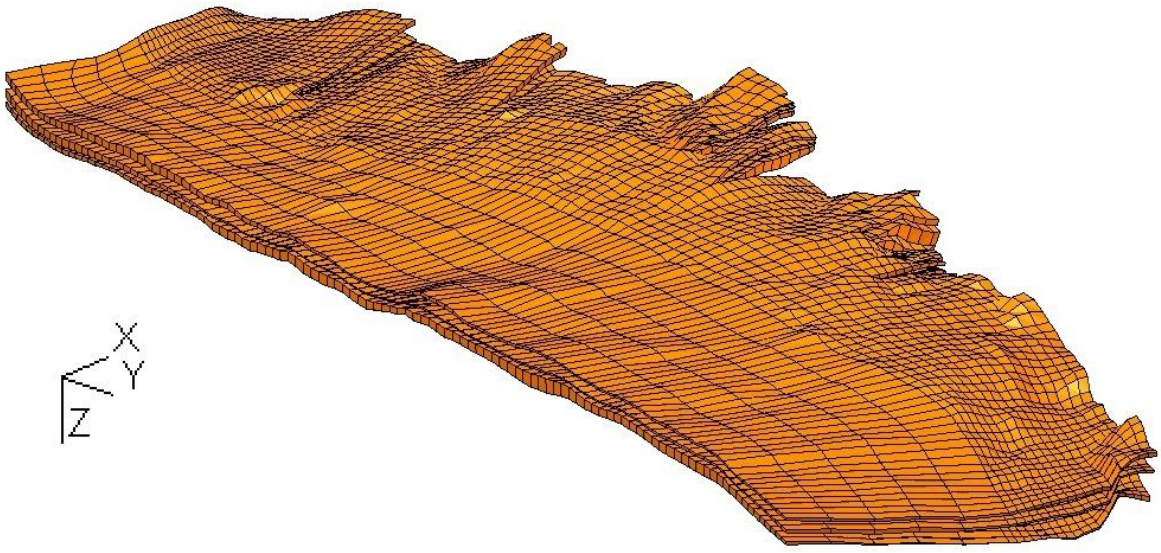
Podíváme-li se na tyto modely, jsou na první pohled patrné zlomy v síti a chybějící buňky. Jedná se tedy o modely, jejichž tvar není spojitý. I u těchto modelů jsem použil vytažení v ose Z a tím jsem dokázal zvýraznit praskliny a členitost povrchu.

- 8. SARMAT

  Velikost sítě: 28 x 109 x 5

  Počet uzlů: 122 080

  Počet elementů celkem / aktivních: 15 260 / 8 050



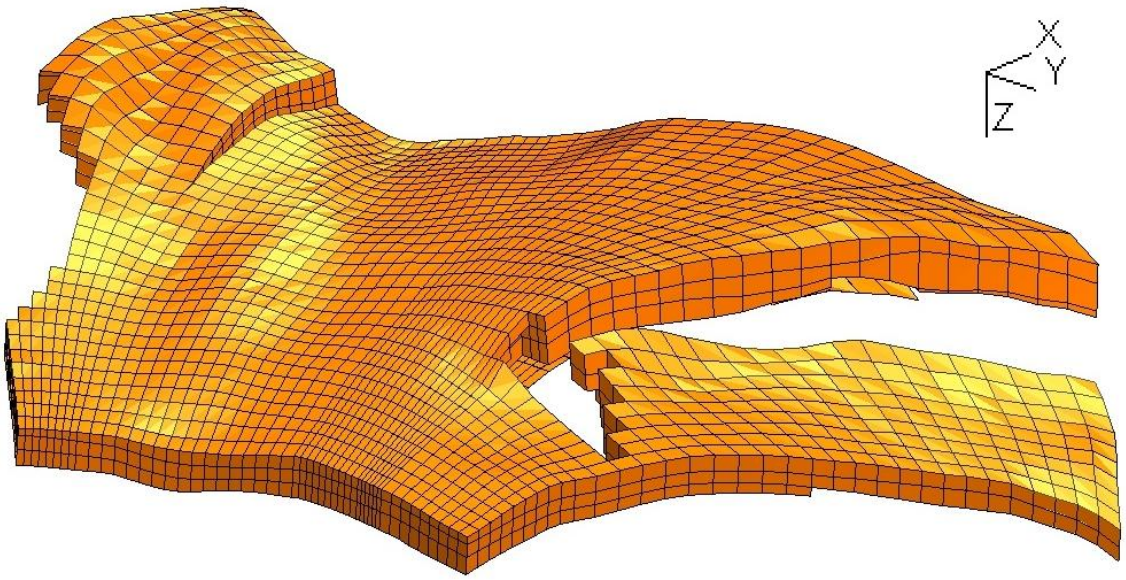
Obr. 5.7 *8. Sarmat - síť*

- 9. BADEN

  Velikost sítě: 33 x 50 x 5

  Počet uzlů: 66 000

  Počet elementů celkem / aktivních: 8 250 / 5 085



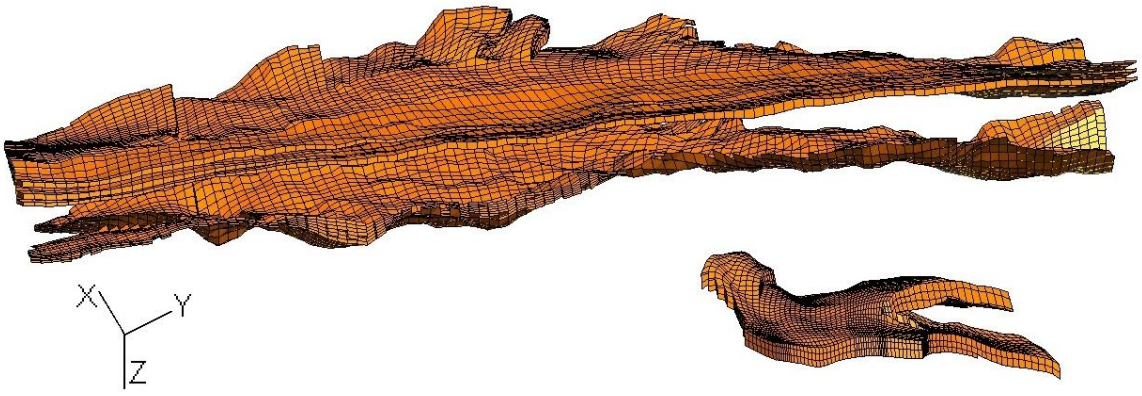
Obr. 5.8 *9. Baden - síť*

- SPOJENÝ MODEL

  Velikost sítě: 87 x 115 x 6

  Počet uzlů: 480 240

  Počet elementů celkem / aktivních: 60 030 / 23 923



Obr. 5.9 *Spojený model – síť*

5.4 Porovnání času výpočtu a využitého datového prostoru

Jak už bylo řečeno, můžeme využít interpretovaného konvertoru a binárního konvertoru. Možnost snadného spuštění zatím vyzdvihuje binární konvertor. Jak jsou na tom s množstvím času, který je potřeba na převod souborů, vidíme v Tabulka 5.1. Je naprosto jasné, že binární konvertor si poradí s převodem mnohem rychleji, především u velkého počtu uzlů. Závislost času převodu na počtu uzlů v síti vidíme v Graf 5.1. Průběh grafu je téměř lineární.

Velikost jednotlivých vstupních a výstupních souborů nalezneme v Tabulka 5.2. Binární EGRID je jednoznačně nejmenší, poté jsou MSH a PRT a největší je soubor POS. Protože má binární soubor slova o velikosti 4B a nejsou zde žádné nepotřebné znaky a informace, dosáhneme tak nejmenší velikosti dat. Oproti tomu MSH, PRT a POS jsou textové formátované soubory, kde jeden znak má 1B až 2B. Je zde spousta bílých znaků a dalších znaků, které nenesou žádnou informaci. Soubor PRT v sobě nemá definici gridu, ale v POS musí být síť nadefinována. Navíc u souboru POS musíme každému rohu elementu přiřadit velikost dané veličiny, ačkoliv je tato hodnota stejná pro celý element. To způsobuje značný nárůst velikosti souboru. Z Graf 5.2 plyne lineární nárůst velikosti souborů na počtu uzlů v síti.

Konfigurace PC, na kterém testy probíhaly:

Intel Core 2Duo 2,13GHz, 3MB L2 Cache, 1066MHz FSB

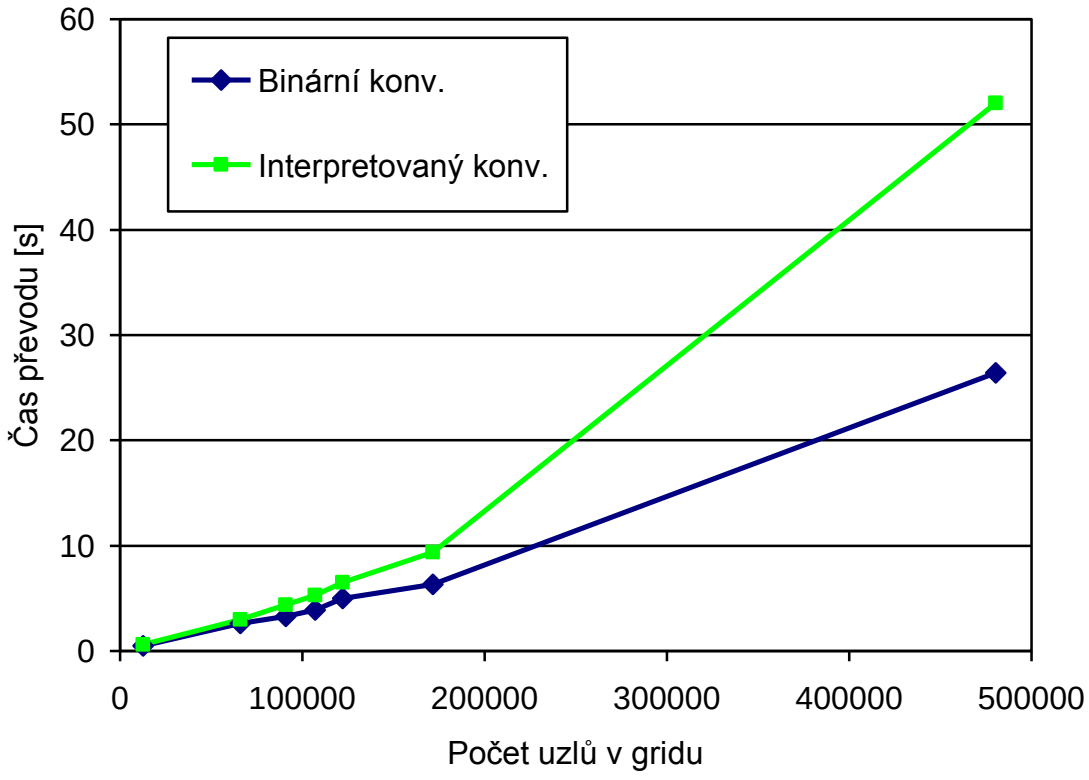
RAM 4GB, 1066MHz

Disk 60GB, 5400 ot./min.

Tabulka 5.1 *Čas převodu souborů*

Název modelu	*Čas převodu [s]*			
	EGRID → MSH		PRT → POS	
	egrid2msh.exe	egrid2msh.pl	prt100pos.exe	prt100pos.pl
Gaswater	0,5	0,6	0,7	0,8
Dunajovice	3,3	4,4	4	5
Lobodice	3,9	5,3	9,9	11
Štramberk	6,3	9,4	7,3	10,3
8. sarmat	5	6,5	---	---
9. baden	2,6	3	---	---
Spojený model	26,4	52	---	---

Graf 5.1 *Závislost času převodu na počtu uzlů v gridu*



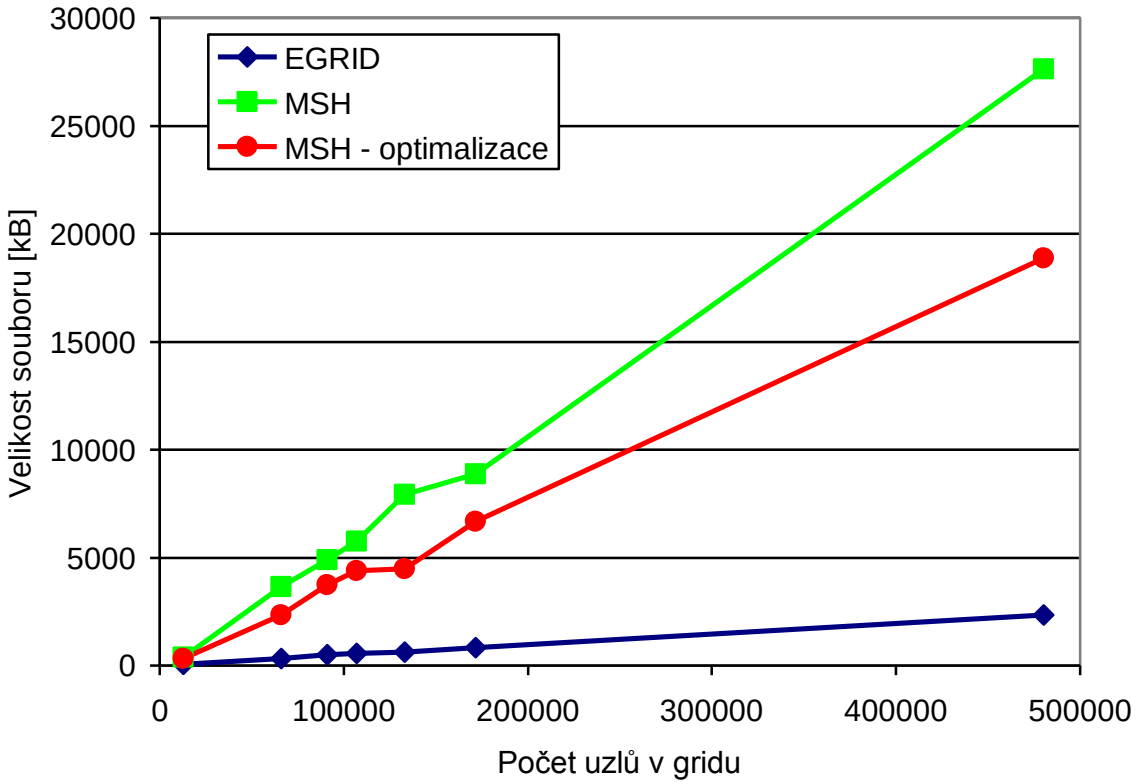
Tabulka 5.2 *Velikost souborů jednotlivých modelů*

Název modelu	*Velikost souboru [kB]*					
	EGRID	MSH	PRT	POS		
				STAT	SGAS	PRESS
Gaswater	68	390	170	---	731	731
Dunajovice	496	4 897	3 919	21 458	---	---
Lobodice	554	5 768	8 678	30 253	29 393	29 340
Štramberk	844	8 883	6 247	29 468	5 971	---
8. sarmat	614	7 913	-----			
9. baden	333	3 645				
Spojený model	2 356	27 626				

Tabulka 5.3 *Velikost souborů po optimalizaci*

Název modelu	*Velikost souboru po optimalizaci [kB]*			
	MSH	POS		
		STAT	SGAS	PRESS
Gaswater	337	---	700	700
Dunajovice	3 726	16 380	---	---
Lobodice	4 394	23 106	28 201	28 149
Štramberk	6 682	22 324	4 542	---
8. sarmat	4 470	-----		
9. baden	2 330			
Spojený model	18 881			

Graf 5.2 *Závislost velikosti souboru na počtu uzlů v síti*



6 Závěr

V závěrečné kapitole nalezneme vyhodnocení této diplomové práce. Také možné směry pokračování a vývoje konvertorů.

6.1 Celkové zhodnocení práce

Ačkoli práci provázely drobné komplikace, podařilo se mi konvertory naprogramovat a zdárně odladit na sadě testovacích úloh (kapitola 5). Největší problém byl s binárním souborem EGRID a sestavením sítě. Přestože jsem měl k systému Eclipse referenční manuál v anglickém jazyce, informace zde obsažené nebyly vždy konkrétní a podrobné.

Nejdříve bych se zmínil o konvertoru egrid2msh. Jedná se o stavební kámen této práce. První a hlavní úkol bylo seznámení s formátem souboru EGRID, aby bylo možné soubor dekódovat a získat z něj potřebné informace. Jelikož je soubor binární, bylo nejdříve nutné data převést do podoby, se kterou by se dalo jednodušeji pracovat. Používám tedy šestnáctkové vyjádření 32-bitových slov, které pomocí dvou podprogramů převádím na typ integer nebo real. V této chvíli stačí pomocí pár algoritmů sestavit rohové uzly a výsledky zapsat do výstupního souboru MSH.

Když byl převod souboru EGRID hotový, mohl jsem pokračovat v konvertoru prt100pos. Ten využívá soubor EGRID pro vytvoření elementů, kterým pak přiděluje hodnoty daných veličin získaných ze souboru PRT. Jelikož PRT je textový soubor, musel jsem hodnoty hledat pomocí klíčových slov a různých značek. Hodnoty simulovaných veličin jsou uloženy pomocí rozdělených vícerozměrných matic. Po vytvoření algoritmu pro přeuspořádání prvků v matici stačilo takto získané hodnoty přiřadit jednotlivým elementům a zapsat do výstupního souboru POS.

Přestože se tato práce měla zabývat konvertorem systému E100, vytvořil jsem i konvertor prt300pos pro systém E300. Soubory EGRID mají stejnou strukturu, stačilo tedy pozměnit získávání hodnot ze souboru PRT.

6.2 Použití a omezení konvertorů

Konvertory slouží k převodu souboru EGRID na MSH a PRT na POS, které můžeme graficky zobrazit pomocí programu Gmsh. Pracují výhradně s corner point geometrií gridu. Výhodou je však funkce vytažení sítě v ose Z. Ta nám umožňuje zvýraznit ploché a nevýrazné modely, ale také snížit příliš členité a nepřehledné sítě. Konvertory vloží informaci o jednotce simulované veličiny a čase simulace do souboru POS. Pokud je ve výstupním souboru PRT více výpisů dané veličiny, zobrazí se ve výsledku jako animace v čase. Konvertor však nepracuje s radiální sítí, kterou by bylo složité z jednotlivých elementů poskládat. Jelikož neexistuje program pro zobrazení výsledků systému Eclipse, který by byl jednoduchý, rychlý a zároveň nekomerční, je tato alternativa velmi vhodná. Ústav nových technologií a aplikované informatiky na TUL využívá Systém Eclipse. Myslím si, že jim tento malý pomocník, v podobě konvertoru, ušetří čas a zjednoduší práci.

6.3 Možnosti dalšího pokračování

Jako další možnost pokračování v této práci je sestavení algoritmu pro tvorbu radiálních sítí. Dále by bylo možné vytvořit grafické rozhraní konvertoru, kde by bylo možné zadávat více vstupních hodnot. Nejen tedy vstupní soubor a vytažení sítě, ale například by bylo možné nadefinovat si vlastní tagy pro elementy v souboru MSH. Další funkce, která by mohla mít přínos, je možnost vykreslení jednotlivých vrstev modelu nebo konkrétních elementů a také volba vypisovaných veličin. Pokud by se pracovalo s modely, kde počet uzlů přesahuje 300tis., určitě by pomohla i další optimalizace kódu. Vhodné by bylo i spojení konvertoru prt100pos a prt300pos, kde by se automaticky detekovalo, o jaký typ souboru jde.

6.4 *Možnosti propojení Eclipse a Flow 123d*

Jedním z úkolů bylo navrhnout možnosti dalšího propojení mezi programem Eclipse a modelovacími nástroji používanými na NTI (systém Flow 123d). Bohužel vývoj a testy konvertorů byly náročnější, než se čekalo. Proto zde danou problematiku pouze naznačím.

Síť

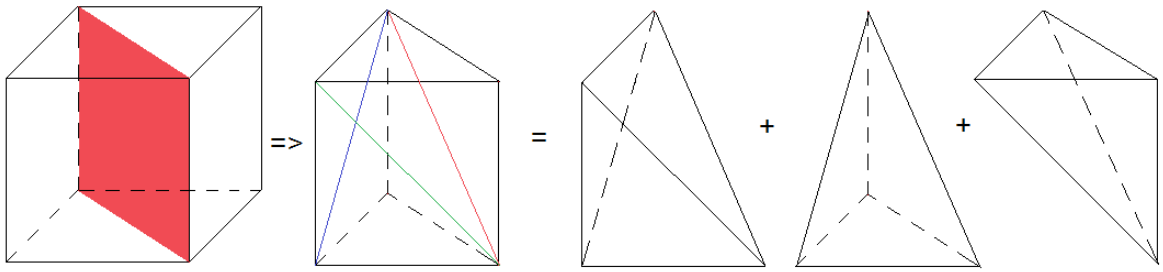
Flow 123d používá čtyřstěny pro sestavení sítě na rozdíl od Eclipse, který používá šestistěny. Museli bychom rozbít každý šestistěn na šest čtyřstěnů, tak aby nám souhlasily hrany sousedních elementů. Problém by nastal u modelů, kde dochází ke zlomům.

Materiálové vlastnosti

Museli bychom vytvořit konvertor, který by načetl hodnoty (PORO, PERM) ze souboru PRT a převedl do formátu souboru MTR, který používá Flow 123d. Následně bychom museli přiřadit odpovídajícím elementům v MSH id číslo materiálu ze souboru MTR.

Výsledky

Výsledky simulací jsou uloženy v souboru POS. Porovnávali bychom vypočtené hodnoty v čtyřstěnech s šestibokými bloky.



Obr. 6.1 *Rozbití šestistěnu na čtyřstěny*

Literatura

[1] **Schlumberger Informations Solutions:** *Dokumentace systému Eclipse*, 2007

[2] **GEUZAINE, Ch**; **REMACLE, J. F.:** *GMSH reference manual*, 2009

[3] **KOLÁŘ, M.:** *Testování modelu proudění podzemních vod na konformních a nekonformních sítích*, Bakalářská práce, TU Liberec, 2007

[4] ***CPAN.CZ*** [online], 2001, Jazyk Perl, Dostupné z WWW: <http://www.perl.cz/>

[5] ***Comprehensive Perl Archive Network*** [online], © 1995-2010 Jarkko Hietaniemi, Dostupné z WWW: <http://www.cpan.org/>

[6] ***Gmsh*** [online], 1997, A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities, Dostupné z WWW: <http://geuz.org/gmsh/>

Přílohy

Obsah přiloženého CD

- egrid2msh.pl
- prt100pos.pl
- prt300pos.pl
- egrid2msh.exe
- prt100pos.exe
- prt300pos.exe
- Gmsh.exe – program Gmsh
- binární konvertory přeložené pomocí pl2bin
- optimalizované modely
- modely bez optimalizace
- DP_Kolar_Michal.pdf